Polycopié de cours et exercices corrigés

INTRODUCTION A LA LOGIQUE MATHEMATIQUE

Support destiné aux étudiants de la 1ère année PES (Professeur d'Enseignement Secondaire) informatique, Licence Mathématique et Informatique (MI)

Dr. BENKADDOUR Fatima Zohra
Enseignante au département des sciences exactes
E.N.S d'Oran

Année universitaire 2018-2019

Résumé

Cet ouvrage propose une introduction à la logique mathématique accessible aux étudiants de la première année PES (Professeur d'enseignement Secondaire) à l'ENS d'Oran (E.N.S.O) ainsi qu'aux étudiants du domaine Mathématique et Informatique (MI).

Il aborde les techniques fondamentales de la logique propositionnelle dite d'ordre 0 et la logique des prédicats du premier ordre. Ce document propose une série d'exercices résolus qui conduisent l'étudiant à une connaissance approfondie des notions de base de la logique.

Mots-clés : *Logique propositionnelle, Logique des prédicats, PES, E.N.S.O, MI.*

Table des matières

Résumé ii

Table des matières iii

Liste des figures vi

Liste des tableaux vii

1 Notions de base de la logique mathématique 3

1.1 Introduction . . . 3
1.2 Assertion . . . 3
1.3 Tautologies et antilogies . . . 4
1.4 Connecteurs . . . 4
1.4.1 La négation « non » ou « $\neg$ » . . . 4
1.4.2 Conjonction « et » ou « $\wedge$» . . . 5
1.4.3 Disjonction « ou » ou « $\vee$ » . . . 6
1.4.4 Implication « $\Rightarrow$ » . . . 7
1.4.5 Équivalence « $\Leftrightarrow$ » . . . 8
1.5 Lois de Morgan . . . 9
1.6 Les quantificateurs . . . 11
1.6.1 Le quantificateur universel . . . 11
1.6.2 Le quantificateur existentiel . . . 12
1.6.3 Propriétés des quantificateurs . . . 12
1.7 Théorie des ensembles . . . 13
1.7.1 Ensemble . . . 13
1.7.2 Élément . . . 14
1.7.3 Sous-ensemble . . . 14
1.7.4 Complémentaire d'un ensemble . . . 14
1.7.5 Intersection de deux ensembles . . . 14
1.7.6 Union de deux ensembles . . . 15
1.8 Relations . . . 16
1.8.1 Relation d'équivalence . . . 16
1.8.2 Relation d'ordre . . . 16
1.9 Ensembles ordonnés et Treillis . . . 17
1.9.1 Ensembles ordonnés . . . 17

1.9.2 Les Treillis 18
1.10 Conclusion 18

2 Logique propositionnelle (Ordre 0) 19
2.1 Introduction 19
2.2 Syntaxe (Formalisation) 19
2.3 Sémantique 21
2.3.1 Table de vérité 21
2.3.2 Validité et consistance 22
2.3.3 Équivalence des formules bien formées 22
2.3.4 Conséquence logique (satisfaction) 23
2.3.5 Formes normales (Normalisation) 24
2.4 Théorie de la preuve (démonstration) 25
2.4.1 Système de déduction naturelle 27
2.4.1.1 Règles d'inférence 28
2.4.1.2 Complétude et consistance 28
2.4.2 Principe de résolution de Robinson 28
2.4.2.1 Règle de résolution 29
2.4.2.2 Validité de la règle de résolution 30
2.4.2.3 Complétude de la méthode de résolution 31
2.4.2.4 Quelques stratégies 31
2.5 Conclusion 36

3 Logique des prédicats (ordre 1) 37
3.1 Introduction 37
3.2 Définitions 37
3.2.1 Prédicat 37
3.2.2 Formule 38
3.2.3 Formule atomique 39
3.2.4 Formule bien formée 39
3.2.5 Variables liées, variables libres 39
3.3 Théorie des modèles 41
3.3.1 Interprétation 41
3.3.1.1 Interprétation des termes 41
3.3.1.2 Interprétation des formules 41
3.3.2 Validité et consistance 42
3.3.3 Équivalence des formules bien formées 43
3.3.3.1 Formules équivalentes 43
3.3.3.2 Forme prénexe 43
3.3.4 Forme de Skolem 45
3.3.5 Forme Clausale 47
3.4 Conclusion 49

A Série d'exercices **50**

B Corrigés des exercices **57**

C Examen corrigé **63**

Ouvrages de références **67**

Table des figures

1.1 Exemple d'ensemble ordonné 17
1.2 Solution de l'exemple d'ensemble ordonné 18

2.1 Complétude de la méthode de résolution 31

B.1 Table de vérité de la formule A 60
B.2 Table de vérité des formules propositionnelles 61

Liste des tableaux

1.1 Table de vérité de la négation 4
1.2 Table de vérité de la conjonction 6
1.3 Table de vérité de la disjonction 7
1.4 Table de vérité de la disjonction 8
1.5 Table de vérité de l'équivalence 9
1.6 Exemple de l'équivalence 10

2.1 Exemple de table de vérité réduite 21
2.2 Exemple de la conséquence valide 24

A.1 Table des formules 52
A.2 Table des formules à simplifier 52
A.3 Table des formules à vérifier 52
A.4 Table des variables 55
A.5 Table des prédicats 56

B.1 Simplification des formules 59

C.1 Table de vérité $(p|q)$ 65
C.2 Table de vérité $(p|q)|(p|q)$ 65
C.3 Table de vérité $(p|p)$ 65
C.4 Table de vérité de P 66

Avant-propos

Ce présent document se veut un support de cours pour les étudiants de la 1re année PES (Professeur d'enseignement Secondaire) à l'École Normale Supérieure d'Oran (E.N.S.O) [1].
Néanmoins, il décrit aucune spécificité propre à l'environnement informatique de l'ENSO et pourra s'avérer utile à tout étudiant désirant apprendre les bases de la logique mathématique.
Ce document n'est pas exhaustif et survole les notions de base de la logique qui sont généralement les plus utiles.
La structure et le contenu des chapitres de ce document sont synchronisés avec le nouveau contenu du programme établi dans le canevas de la formation. Il est organisé au tour de trois (03) chapitres et trois (03) annexes.

— Le premier chapitre intitulé *Notions de base de la logique mathématique* introduit les notions fondamentales de la logique mathématique qui s'avèrent indispensables aux chapitres suivants.
— Le deuxième chapitre appelé *Logique propositionnelle (d'ordre 0)* ou *Calcul propositionnel* est la première étape dans la construction du calcul des prédicats. Il permet de définir les règles de déduction qui relient les propositions entre elles, sans en examiner le contenu.
— Le troisième chapitre aborde la notion de *Logique des prédicats (d'ordre 1)*. La logique du premier ordre est considérée par nature plus expressive que la logique des propositions, et permet de représenter des connaissances relatifs à des environnements complexes. Elle est construite à partir de la logique propositionnelle et s'inspire du langage naturel.
— En annexes, nous proposons une série d'exercices résolus ainsi qu'un examen corrigé.

Ce cours permet aux étudiants de :

- Connaître les principaux opérateurs,quantificateurs universels et leurs propriétés : NON, ET, OU,...
- Comprendre les notions d'implication et d'équivalence ;
- Modéliser un énoncé afin de tester sa validité ;
- Appliquer toutes ces notions à la démonstration mathématique ;
- Structurer proprement un raisonnement ;
- Tester ses connaissances à travers une série d'exercices corrigés.

1. www.ens-oran.dz

Chapitre 1

Notions de base de la logique mathématique

1.1 Introduction

La logique est la base fondamentale de tous les raisonnements mathématiques. Elle est très importante pour l'énonciation de propositions et l'étude de leur valeur de vérité.
Dans ce premier chapitre, nous introduirons les bases de la branche des mathématiques appelée logique. Nous présenterons en particulier les définitions d'assertions, de tautologies et d'antilogies ainsi que les différents connecteurs logiques.
Ce chapitre prévoit les notions apparaissant dans les deux chapitres suivants soient la logique propositionnelle et la logique descriptive du premier ordre.

1.2 Assertion

Une assertion est un énoncé mathématique auquel on attribue l'une des deux valeurs logiques : le vrai (V) ou le faux (F).

Exemple

— L'assertion «$1 + 1 = 2$» est vraie.
— L'assertion «$2 + 2 = 5$» est fausse.

— Les propriétés, théorèmes sont des assertions vraies.

1.3 Tautologies et antilogies

Les assertions (dépendantes de P et Q) qui sont vraies quelle que soit la valeur de vérité de P et Q sont dites des tautologies. Une tautologie est en fait un théorème de logique. Les assertions (dépendantes de P et Q) qui sont fausses quelle que soit la valeur de vérité de P et Q sont dites des antilogies.

1.4 Connecteurs

Ils existent cinq (5) connecteurs logiques, à la base de tout raisonnement mathématique. Soient P et Q deux assertions.

1.4.1 La négation « non » ou « ¬ »

Nous appellerons la négation de P, l'assertion (non P)(not P) et qui sera notée sous forme formalisée ¬P ou $\bar{P}$.

Table de vérité de la négation

Soit la proposition P (voir la table 1.1). La négation d'une proposition P (vraie)

P	$\neg P$
1	0
0	1

TABLE 1.1: Table de vérité de la négation

est une proposition fausse. Si P (fausse) alors $\neg P$ est vraie.

Négation de la négation

En général, une double négation vient souvent renforcer la négation tel que : voulez-vous sortir ? non, non.
En mathématique, une double négation est considérée comme une affirmation.

Exemples

1. Si P est la proposition $x = 0$, $\neg P$ est la proposition $x \neq 0$.
2. Le 05 est non pair donc 05 est impair.

Remarque

Le sens du symbole $\Longleftrightarrow$ qui se lit équivaut, et qui signifie ici que les deux propositions ont toujours la même valeur.

1.4.2 Conjonction « et » ou « $\wedge$»

Nous appellerons conjonction de P et Q, l'assertion (P et Q) (P and Q) et qui sera notée P $\wedge$ Q.

Exemple

P : « La terre est ronde »(vraie) et Q : « Le ciel est bleu » (vraie).
P et Q ou P $\wedge$ Q se lit donc « La terre est ronde **ET** le ciel est bleu ». P $\wedge$ Q est vraie. Nous dirons que l'assertion P $\wedge$ Q est fausse lorsque l'une au moins des deux assertions est fausse. Ainsi « La terre est ronde **ET** le ciel est vert » est une assertion fausse.

Commutativité

(P et Q) $\Longleftrightarrow$ (Q et P)

Table de vérité de la conjonction

Le résultat de la conjonction est démontré par la table de vérité 1.2.

P	**Q**	$P \wedge Q$
1	1	1
1	0	0
0	1	0
0	0	0

TABLE 1.2: Table de vérité de la conjonction

1.4.3 Disjonction « ou » ou « $\vee$ »

Nous appellerons disjonction de P et Q, l'assertion (P ou Q) et qui sera notée P $\vee$ Q.

Remarque

En mathématiques, le «ou» est non-exclusif, c'est à dire qu'il comprend la possibilité que les deux propositions soient vraies. Ainsi la proposition « $xy = 0$ » équivaut à la proposition « $x = 0$ ou $y = 0$ » , elle est vraie quand l'un des deux nombres est nul, elle est aussi vraie quand les deux sont nuls.

Commutativité

$(P \text{ ou } Q) \Longleftrightarrow (Q \text{ ou } P)$

Table de vérité de la disjonction

Le résultat de la disjonction est démontré par la table de vérité 1.3.

P	**Q**	$P \vee Q$
1	1	1
1	0	1
0	1	1
0	0	0

TABLE 1.3: Table de vérité de la disjonction

1.4.4 Implication « ⇒ »

La proposition notée « P ⇒ Q » correspond à la proposition $NonP$ ou Q. P s'appelle alors l'hypothèse et Q la conclusion. $P \Rightarrow Q$ est une proposition qui se nomme implication et que nous pouvons lire de différentes façons :

— Si P alors Q,
— Pour que P il faut Q,
— Pour que Q il suffit P,
— P est une condition suffisante pour Q,
— Q est une condition nécessaire de P.

Table de vérité de l'implication

Le résultat de l'implication est montré à la table 1.4.

P	**Q**	$\neg P$	**P** $\Rightarrow$ **Q**
1	1	1	1
0	1	1	1
1	0	0	0
0	0	1	1

TABLE 1.4: Table de vérité de la disjonction

L'assertion est vraie dès lors que P est fausse (quelle que soit la vérité de Q). Si P est vraie et (P $\Rightarrow$ Q) vraie alors Q est vraie. De plus l'implication Q $\Rightarrow$ P s'appelle la réciproque de l'implication P $\Rightarrow$ Q.

Exemples

— $P : 2 = 2$ et $Q : 4 = 4$ sont deux assertions vraies, donc $P \Rightarrow Q$ ou ($NonP$ ou Q) est vraie,
— Si $x \in \{5, 6, 9\}$ alors $x \leqslant 9$ est une assertion vraie.

1.4.5 Équivalence « $\Leftrightarrow$ »

Nous dirons que deux assertions sont logiquement équivalentes si elles ont la même valeur de vérité et seront notées $P \Leftrightarrow Q$. En d'autres termes $P \Leftrightarrow Q$ est vraie si P et Q sont toutes les deux vraies ou si toutes les deux sont fausses. La proposition $P \Leftrightarrow Q$ correspond à la proposition $(P \Rightarrow Q)$ et $(Q \Rightarrow P)$. Nous pourrons l'exprimer comme suit :

— P est équivalent à Q,
— Pour P, il faut et il suffit Q,
— P est une condition nécessaire et suffisante pour Q,

— P si et seulement si Q.

Table de vérité de l'équivalence

Le résultat de l'équivalence est montré à la table 1.5.

P	**Q**	$P \Leftrightarrow Q$
1	1	1
0	1	0
1	0	0
0	0	1

TABLE 1.5: Table de vérité de l'équivalence

Exemple

Prenons (P et Q) et (Non (Non P ou Non Q)). Voir le résultat à la table 1.6.

1.5 Lois de Morgan

Les lois de De Morgan permettent de transformer une conjonction en une disjonction (et réciproquement) via la négation.

P	**Q**	**Non P**	**Non Q**	**Non P $\vee$ Non Q**	**Non(Non P ou Non Q)**	**P $\wedge$ Q**
V	V	F	F	F	V	V
F	V	V	F	V	F	F
V	F	F	V	V	F	F
V	F	F	V	V	F	F

TABLE 1.6: Exemple de l'équivalence

Remarque

La méthode des tables de vérité fournit une preuve de ces propriétés. Parmi les règles de calcul logique importantes il y a aussi les règles suivantes.

1. **Règle de double-négation :**

$$\bar{\bar{A}} = A$$

2. **Règle de commutativité :**

$$\mathrm{A} * \mathrm{B} = \mathrm{B} * \mathrm{A}$$

avec $* = .$ ou $* = +$

3. **Règle de distributivité :**

$$\mathrm{A} + (\mathrm{B} \,.\, \mathrm{C}) = (\mathrm{A} + \mathrm{B}) \,.\, (\mathrm{A} + \mathrm{C})$$

et

$$\mathrm{A} \,.\, (\mathrm{B} + \mathrm{C}) = (\mathrm{A} \,.\, \mathrm{B}) + (\mathrm{A} \,.\, \mathrm{C})$$

4. **Règle d'associativité :**

$$\mathrm{A} * (\mathrm{B} * \mathrm{C}) = (\mathrm{A} * \mathrm{B}) * \mathrm{C}$$

5. **Règle d'idempotence :**

$$A * A = A$$

avec $* = .$ ou $* = +$

Toute expression booléenne peut être écrite sous une forme particulière.

1. **Forme normale conjonctive (FNC) :** Une expression logique est en FNC si et seulement si elle est une conjonction d'une ou plusieurs disjonctions d'un ou plusieurs littéraux tel que : $(A + \bar{B} + \bar{C}).(\bar{D} + E + F)$
2. **Forme normale disjonctive (FND) :** Une expression logique est en FND si et seulement si elle est une disjonction d'une ou plusieurs conjonctions d'un ou plusieurs littéraux. Par exemple,$(\bar{A}.\bar{B}) + B + (\bar{C}.D.E)$; sachant qu'un littéral est une variable booléenne (une lettre) ou la négation d'une variable.

1.6 Les quantificateurs

Nous considérons dans ce qui suit deux types de quantificateurs.

1.6.1 Le quantificateur universel

Un quantificateur permet de préciser le domaine de validité d'une proposition. Le symbole $\forall$ qui signifie « quel que soit » ou « pour tout » représente le quantificateur universel. Ce symbole représente la lettre « A » renversée qui est l'initiale du mot anglais « All ». Il doit toujours être suivi du signe d'appartenance $\in$.

Exemple

$$\forall x \in R, x^2 \geq 0$$

Qui signifie « quel que soit x appartenant à R, x^2 est positif ou nul ».

1.6.2 Le quantificateur existentiel

Le symbole $\exists$ qui signifie « il existe au moins un ... tel que » représente le quantificateur existentiel. Ce symbole représente la lettre « E » renversée qui est l'initiale du mot anglais « exist ». On peut éventuellement rajouter un point d'exclamation pour montrer l'unicité. On a alors : $\exists!$ qui signifie « *il existe un unique...tel que*».

Exemple

$$\exists! x \in [0,1], x^2 + 4x + 1 = 0$$

Qui signifie « Il existe un unique x appartenant à l'intervalle $[0,1]$ tel que : $x^2 + 4x + 1 = 0$ ».

1.6.3 Propriétés des quantificateurs

Les quantificateurs possèdent un certain nombre de propriétés dont :

— **L'ordre des quantificateurs :** L'ordre dans lequel on écrit les quantificateurs change la signification.

Exemple : $\forall x \in R, \exists y \in R, y > x$, qui signifie « *Quel que soit le réel x, il existe au moins un réel y tel que y soit supérieur à x* ». On peut toujours trouver un nombre supérieur à un nombre réel donné car l'ensemble R n'est pas borné. La proposition est vraie.

Inversons maintenant les quantificateurs $\exists x \in R, \forall y \in R, y > x$, « *Il existe au moins un réel x tel que pour tout réel y, y soit supérieur à x* ». Cette proposition cette fois est fausse car on ne peut trouver un réel inférieur à tous les autres. En effet l'ensemble R n'a pas de borne inférieure.

— **Négation d'une proposition universelle :**Une proposition universelle s'énonce : « *Pour tout élément x d'un ensemble E, x possède la proposition P* ». Sa négation sera : « *il existe au moins un élément x de l'ensemble E qui ne possède pas la propriété P* ».

Exemple : soit la proposition « *Tous les lecteurs de ce chapitre comprennent tout ce qui est écrit*» sa négation sera donc : « *Il existe au moins un lecteur qui ne comprend pas ce chapitre* ».

Remarque : Pour démontrer qu'une proposition universelle n'est pas vraie,

il suffit donc de trouver un seul x qui ne vérifie pas la proposition P. C'est ce qu'on nomme un « *contre-exemple* ».

— **Négation d'une proposition existentielle :**Une proposition existentielle s'énonce : « *Il existe au moins un élément x de l'ensemble E qui possède la propriété P* ». Sa négation sera : « *Pour tout élément x de l'ensemble E, x ne vérifie pas P* ».

Exemple : Soit la proposition P : $\exists x \in R, x^2 = -1$. Cette proposition est fausse car un carré ne peut être négatif. Par contre sa négation est vraie : $\forall x \in R, x^2 \neq -1$.

1.7 Théorie des ensembles

La théorie des ensembles se donne comme primitives les notions d'ensemble et d'appartenance, à partir desquelles elle reconstruit les objets usuels des mathématiques : fonctions, relations, etc. Dans ces sous-section, nous présenterons les notions d'ensemble, de sous-ensemble et de relation.

1.7.1 Ensemble

Un ensemble est une collection d'éléments que l'on peut énumérer ou définir par une propriété. Certains ensembles ont des notations particulières (ex. N, Z, D, Q, R). Lorsqu'on énumère les éléments d'un ensemble, on dit que cet ensemble est défini par extension, lorsqu'on définit un ensemble par une propriété, on dit que cet ensemble est défini par compréhension. Un ensemble qui ne contient aucun élément s'appelle : l'ensemble vide noté $\emptyset$.

Exemple

N ensemble des entiers naturels, $N = \{1, 2, 3, ...\}$
Lorsque le nombre des éléments d'un ensemble devient trop important ou qu'il y a un nombre infini d'éléments, on ne peut le définir par compréhension tel que : $C = \{x \in N/1 \leqslant x \leqslant 49\}$.

1.7.2 Élément

Un ensemble est constitué d'éléments. On représente souvent un élément par une minuscule. On dit qu'un élément « a » appartient à un ensemble A. On écrit alors : $a \in A$. Le symbole $\in$ signifiant « *appartient à* » est initiale de « *élément* ».

1.7.3 Sous-ensemble

On dit qu'un ensemble A est un sous-ensemble de l'ensemble E si et seulement si tout élément de A est élément de E ou si $A = \emptyset$. On dit alors que A est inclus dans E. $A \subset E \Leftrightarrow \forall a \in A, a \in E$ ou $A = \emptyset$. Le symbole $\subset$ signifie « *inclus dans* ».

1.7.4 Complémentaire d'un ensemble

On appelle complémentaire de l'ensemble A dans l'ensemble E, l'ensemble noté $C_E(A)$ composé des éléments de E qui ne sont pas élément de A. Le complémentaire correspond au connecteur NON. On a alors : $a \in C_E(A) \Leftrightarrow a \in E$ et $a \notin A$. Le symbole $\notin$ signifie « *n'appartient pas à* ». Lorsque l'ensemble$E$ est implicite, on note le complémentaire de A : $\bar{A}$ qui se prononce « *A barre* ».

Exemple

Soit E l'ensemble des entiers naturels inférieurs ou égaux à 20 et soit A l'ensemble des nombres premiers inférieurs à 20. On a donc : $A = \{2, 3, 5, 7, 11, 13, 17, 19\}$.
L'ensemble $C_E(A)$ sera donc l'ensemble des entiers naturels inférieurs ou égaux à 20 qui ne sont pas premiers. On a donc : $C_E(A) = \{0, 1, 4, 6, 8, 9, 10, 12, 14, 15, 16, 18, 20\}$.

1.7.5 Intersection de deux ensembles

On appelle intersection de deux sous-ensembles A et B dans un ensemble E, l'ensemble noté : $A \cap B$ (A inter B) constitué des éléments communs à A et B. On a donc : $x \in A \cap B \Leftrightarrow x \in A$ et $x \in B$.

Exemple

Soit A l'ensemble des entiers naturels pairs inférieurs ou égaux à 20 et soit B l'ensemble des entiers naturels multiples de 3 inférieurs ou égaux à 20.
On a donc :

- $A = \{0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20\}$,
- $B = \{0, 3, 6, 9, 12, 15, 18\}$,
- $A \cap B = \{0, 6, 12, 18\}$ qui n'est autre que l'ensemble des entiers naturels multiples de 6 inférieurs ou égaux à 20.

Remarques

- Lorsque l'ensemble A est inclus dans l'ensemble B, on a alors : $A \subset B \Rightarrow A \cap B = A$
- Lorsque les ensembles A et B sont disjoints, ils ne possèdent aucun élément commun, leur intersection est donc vide, on a donc : $A \cap B = \emptyset$. Ce qui signifie que A et B sont disjoints, comme c'est le cas entre l'ensemble A et sont complémentaire $\bar{A}$, on a donc : $A \cap A = \emptyset$.

1.7.6 Union de deux ensembles

On appelle union de deux sous-ensembles A et B dans un ensemble E, l'ensemble noté : $A \cup B$ (A union B) constitué des éléments qui appartiennent à A ou à B (*éventuellement aux deux, le « ou » étant non exclusif*). On peut alors écrire : $x \in A \cup B \Leftrightarrow x \in A$ ou $x \in B$.

Exemple

Soit A l'ensemble des entiers naturels pairs inférieurs ou égaux à 20 et soit B l'ensemble des entiers naturels multiples de 3 inférieurs ou égaux à 20. On a donc :

- $A = \{0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20\}$,
- $B = \{0, 3, 6, 9, 12, 15, 18\}$,
- $A \cap B = \{0, 2, 3, 4, 6, 8, 9, 10, 12, 14, 15, 16, 18, 20\}$.

Remarques

1. Lorsque l'ensemble A est inclus dans l'ensemble B, on a alors : $A \subset B$ $\Rightarrow A \cap B = B$,
2. L'union de l'ensemble A et de son complémentaire $\bar{A}$ donne l'ensemble E, c'est à dire : $A \cap A = E$.

1.8 Relations

Nous considérons dans ce cours deux types de relations citées dans ce qui suit.

1.8.1 Relation d'équivalence

Une relation est une relation d'équivalence si elle est :

— **Symétrique :** $\forall x \in E, \forall y \in E$, x R y $\Rightarrow$ y R x,

— **Réflexive :** $\forall x \in E, xRx$,

— **Transitive :** $\forall x \in E, \forall y \in E, \forall z \in E, (xRy)$ et $(yRz) \Rightarrow xRz$.

Remarques

1. Dans le cas d'une relation d'équivalence, deux éléments en relation sont aussi dits équivalents,
2. Si R est une relation d'équivalence et x est un élément de E, on appelle classe d'équivalence de x l'ensemble des éléments y de E tels que xRy.

1.8.2 Relation d'ordre

Une relation d'ordre est une relation réflexive, anti-symétrique, transitive.

— **Réflexive :** $\forall x \in E, xRx$,

— **Anti-symétrique :** $\forall x \in E, \forall y \in E, (xRyetyRx) \Rightarrow x = y$,

— **Transitive :** $\forall x \in E, \forall y \in E, \forall z \in E, (xRyetyRz) \Rightarrow xRz$.

Remarques

Si R est une relation d'ordre sur E, alors :

— on dit que l'ordre est total si on peut toujours comparer deux éléments de E : pour tous x, $y \in E$, on a xRy ou yRx. Dans le cas contraire, on dit que l'ordre est partiel.

— Si A est une partie de E et M est un élément de E, on dit que M est un majorant de A si, pour tout $x \in A$, on a xRM.

1.9 Ensembles ordonnés et Treillis

1.9.1 Ensembles ordonnés

Représentation graphique d'un ensemble ordonné

Un ensemble est représenté graphiquement que s'il est fini. Les éléments de l'ensemble E seront représentés par des points de sorte que si on a la relation aRb, alors a sera relié à b par une flèche orientée de a vers b (voir la figure 1.1).

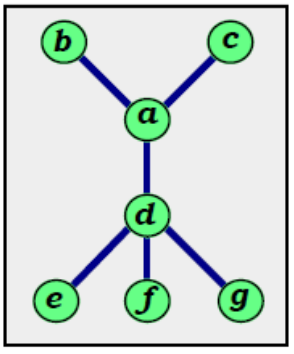


FIGURE 1.1: Exemple d'ensemble ordonné

Exemple

Soit R la relation «*est diviseur de*», $E = \{1, 2, 3, ..., 22, 26\}$ R est transitive (aRb et bRc alors aRc), représentez que les relations directes de type aRb (voir la solution à la figure 1.2).

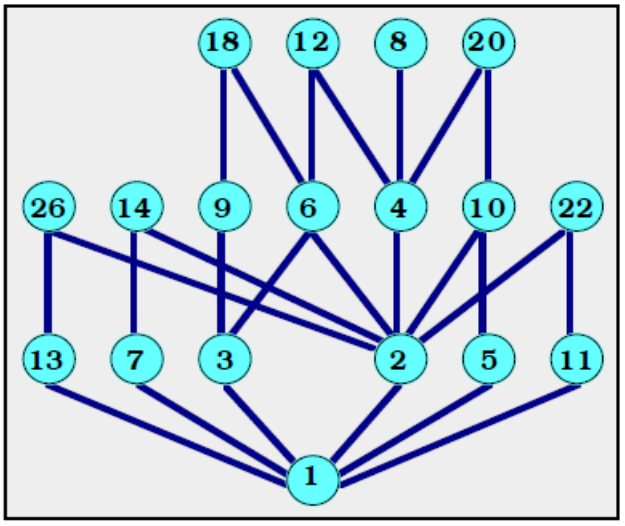


FIGURE 1.2: Solution de l'exemple d'ensemble ordonné

1.9.2 Les Treillis

Un treillis est un ensemble ordonné tel que pour tout couple d'éléments a et b, il existe un plus petit majorant et un plus petit minorant. Le *ppm* noté $a \vee b$ et le *pgm* sera noté $a \wedge b$. Les opérations $\vee$ et $\wedge$ possèdent les propriétés de : commutativité, associativité, idempotence et d'absorption.
L'absorption est définie par : $a \vee (a \wedge b) = a$ et $a \wedge (a \vee b) = a$.

1.10 Conclusion

Dans ce chapitre, nous avons présenté les notions de base de la logique mathématique qui vont nous servir dans la logique descriptive et la logique des prédicats. Nous aborderons dans le prochain chapitre la «*logique d'ordre 0*».

Chapitre 2

Logique propositionnelle (Ordre 0)

2.1 Introduction

Dans la logique propositionnelle, on étudie les relations entre des énoncés, que l'on va appeler propositions ou encore des formules. Ces relations peuvent être exprimées par l'intermédiaire de connecteurs logiques qui permettent, par composition, de construire des formules syntaxiquement correctes. On trouve principalement : la conjonction, la disjonction (inclusive), l'implication, l'équivalence et la négation. Dans ce chapitre, nous présenterons la manière de construire une formule bien formée (fbf) et déterminer par la suite sa valeur de vérité.

2.2 Syntaxe (Formalisation)

S'intéresser à la syntaxe de la logique propositionnelle, c'est considérer les formules qui sont "*bien écrites*". Pour cela, on se donne un alphabet (langage que l'on notera Ls), i.e. un ensemble de symboles, avec :

— **Un ensemble** $V = \{p, q, r, ...\}$ **:** dénombrable de lettres appelées variables propositionnelles. Il s'agit des propositions atomiques telles que par exemple « 6 est divisible par 2 ».

— **Atomes :** Nous appellerons atomes ou variables propositionnelles ou propositions élémentaires des énoncés dont nous ne connaissons pas la structure interne, et qui gardent leur identité tout au long du calcul propositionnel

qui nous occupe. L'ensemble des variables propositionnelles est noté $v(L)$. Elles sont écrites en minuscules $(p, q\dots)$.

— **Un ensemble (fini) de connecteurs logiques :** $\wedge, \vee, \neg, \longrightarrow, \equiv$.

— **Formules :** Nous dénoterons les formules par des lettres majuscules de l'alphabet latin ou grec $(A, B, ...$ ou $\phi\dots)$. L'ensemble des formules, noté $F(L)$, est défini par : les atomes sont des formules $(v(L) \subseteq F(L))$;

Si A et B sont des formules, alors $(A \longleftrightarrow B)$, $(A \longrightarrow B)$, $(A \wedge B)$, $(A \vee B)$ et $(\neg B)$ sont des formules.

Les parenthèses sont un moyen de lever l'ambiguïté. Il en existe un autre qui consiste à donner à chaque opérateur un ordre de priorité. Les connecteurs sont traditionnellement classés de la façon suivante (par priorité décroissante des connecteurs) : $\neg, \wedge, \vee, \longrightarrow, \longleftrightarrow$. Dans le cas où deux connecteurs ont même priorité, et en l'absence de parenthèses, l'associativité se fait de gauche à droite. On peut ainsi se permettre d'omettre les parenthèses. Par exemple, la formule $p \longrightarrow q \longleftrightarrow \neg r$ doit se lire $((p \rightarrow q) \longleftrightarrow (\neg r))$. A l'avenir, lorsque nous parlerons de formules bien formées, nous inclurons également les formules dont le parenthèsage est partiellement ou complètement implicite. L'ensemble des formules bien formées ainsi défini forme le langage de la logique propositionnelle.

Parmi les mots que l'on peut écrire avec cet alphabet, on va regarder ceux qui correspondent à des expressions logiques bien formées que l'on définit (inductivement) ainsi :

1. Toutes les propositions atomiques $p, q, r, ...$, sont des expressions bien formées ;
2. Si A est une expression bien formée, alors $\neg A$ est une expression bien formée ;
3. Si A et B sont deux expressions bien formées, alors $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$ et $(A \equiv B)$ sont des expressions bien formées ;
4. Il n'y a pas d'autres expressions bien formées que celles des règles précédentes. Par exemple $(((\neg p \equiv q) \vee \neg(r \wedge s)) \rightarrow q)$ est une expression bien formée, tandis que $p\neg qr \rightarrow t(\equiv$ ne l'est pas. Dans la suite, on ne considère que des expressions bien formées.

   Littéral : Un littéral est un atome (littéral positif) ou la négation d'un atome (littéral négatif).

   Clause : Une clause est une disjonction de littéraux $(p_1 \vee p_2 \vee ... \vee p_n)$, les littéraux pouvant être positifs ou négatifs.

2.3 Sémantique

S'intéresser à la sémantique de la logique propositionnelle, c'est déterminer la valeur de vérité d'un énoncé, c'est-à-dire d'une formule. On parle de l'interprétation d'une formule : il s'agit plus concrètement d'affecter une valeur vraie ou fausse à chacune des variables propositionnelles qui la compose. Pour une formule à n variables, il y a 2^n mondes possibles.

- **Valuation :** On appelle valuation, ou L-Modèle, d'un ensemble de variables propositionnelles $v \subseteq v(L)$, une fonction m de $v(L)$ dans $\{T, F\}$ ($m : v(L) \longrightarrow \{T, F\}$).
  - $v(\neg X) = \neg v(X)$,
  - $v(x \vee Y) = v(x) \vee v(Y)$,
  - $v(x \wedge Y) = v(x) \wedge v(Y)$,
  - $v(X \supset Y) = v(x) \supset v(Y)$.
- **Interprétation :** Une valuation appliquée à une formule dans laquelle est dite interprétation. soit la formule $F = (a \wedge b) \vee \neg b \longrightarrow \neg a$, $\{m(a) = T, m(b) = F\}$ est une interprétation de F.
- **Modèle d'une formule :** Une interprétation I est un modèle d'une formule A si elle est vraie (si elle vaut **T**).

2.3.1 Table de vérité

On peut tirer partie de ce que dans certains cas, la connaissance de la valeur de vérité d'une sous-formule peut permettre de déterminer la valeur de la formule. Par exemple si $v(B) = 1$, on sait sans connaître $v(A)$ que $v(A \longrightarrow B) = 1$. On peut donc éviter d'énumérer tous les cas (cependant il devient moins évident que tous les cas sont énumérés et cela doit être vérifié).

A	B	C	$A \wedge B$	$(A \wedge B) \longrightarrow C$	$B \longrightarrow C$	$A \longrightarrow (B \longrightarrow C)$
?	?	1	?	1	1	1
0	?	0	0	1	?	1
1	0	0	0	1	1	1
1	1	0	1	0	0	0

TABLE 2.1: Exemple de table de vérité réduite

On vérifie facilement que tous les cas sont bien énumérés (le " ?" désigne 0 ou 1). On a donc montré que $(A \wedge B) \longrightarrow C \equiv A \longrightarrow (B \longrightarrow C)$.

2.3.2 Validité et consistance

- **Formule valide :** Une formule valide, ou tautologie, est une formule Φ vraie quelles que soient les valeurs de vérité des atomes qui la composent (vraie dans toute interprétation). On la note $\models \Phi$.
- **Formule insatisfiable (falsifiable ou inconsistante) :** Une formule insatisfiable, ou sémantiquement inconsistante, ou encore antilogie, est une formule fausse dans toute interprétation.
- **Formules satisfiables (consistante) :** Une formule satisfiable ou sémantiquement consistante est une formule vraie dans au moins une interprétation.
- **Formules invalides :** Une formule invalide est fausse dans au moins une interprétation.
- **Formules contingentes :** Une formule contingente est vraie dans certaines interprétations et fausse dans d'autres.
- **Conséquence valide :** Soient deux formules A et B. Nous dirons que A est la conséquence valide de B (notée $A \models B$) si tout modèle de A est un modèle de B.

2.3.3 Équivalence des formules bien formées

Deux formules sont équivalentes quand elles ont une même valeur dans toutes interprétation (notation : $A \equiv B$).

Théorème : Règle de substitution uniforme

Soit la formule ϕ contenant les atomes $p_1, p_2, ..., p_n$. Soit la formule ϕ^* obtenue en substituant aux atomes $p_1, p_2, ..., p_n$ les formules $1, 2, ..., n$. Alors si $\models \phi$, on $a \models \phi^*$.

Propriétés des formules équivalentes

Soient A, B et C trois formules bien formées.

- $\models [(A \wedge B) \equiv (B \wedge A)]$ par commutativité
- $\models [(A \vee B) \equiv (B \vee A)]$ par commutativité
- $\models [(A \equiv B) \equiv (B \equiv A)]$ par commutativité

- $\models [((A \wedge B) \wedge C) \equiv ((A \wedge (B \wedge C))]$ par associativité
- $\models [((A \vee B) \vee C) \equiv ((A \vee (B \vee C))]$ par associativité
- $\models [((A \equiv B) \equiv C) \equiv ((A \equiv (B \equiv C))]$ par associativité
- $\models [((A \wedge B) \vee C) \equiv ((A \vee C) \wedge (B \vee C))]$ par distributivité
- $\models [((A \vee B) \wedge C) \equiv ((A \wedge C) \vee (B \wedge C))]$ par distributivité
- $\models [A \wedge (A \vee B)) \equiv A$ par absorption
- $\models (A \vee (A \wedge B)) \equiv A]$ par absorption
- $\models [(A \wedge A) \equiv A]$ par idempotence
- $\models [(A \vee A) \equiv A]$ par idempotence
- $\models [\neg(A \wedge B) \equiv (\neg A \vee \neg B)]$ par la lois de Morgan
- $\models [\neg(A \vee B) \equiv (\neg A \wedge \neg B)]$ par la lois de Morgan
- $\models [(A \longrightarrow B) \equiv (\neg A \wedge B)]$ par implication
- $\models [(A \longrightarrow B) \equiv \neg(A \wedge \neg B)]$ par implication
- $\models [(A \equiv B) \equiv ((A \longrightarrow B) \wedge (B \longrightarrow A))]$ par équivalence
- $\models [(A \equiv B) \equiv ((A \wedge B) \vee (\neg A \wedge \neg B))]$ par équivalence
- $\models [(A \longrightarrow B) \equiv (\neg B \longrightarrow \neg A)]$ par contraposition
- $\models [(A \longrightarrow (B \longrightarrow C)) \equiv ((A \longrightarrow B) \longrightarrow (A \longrightarrow C))]$ autodistributivité
- $\models [(A \longrightarrow (B \longrightarrow C)) \equiv (B \longrightarrow (A \longrightarrow C))]$ par import-export
- $\models [((A \longrightarrow B) \wedge (B \longrightarrow C)) \equiv (A \longrightarrow C)]$ par transitivité

2.3.4 Conséquence logique (satisfaction)

On dit que E est une **conséquence valide** de $A, B, \ldots$ si dans la table de vérité de E est vraie à chaque fois que $A, B, \ldots$ sont vraies.
Notée : $A, B, \ldots \models E$.
Pour vérifier que E est une conséquence valide, il faut :

- Trouver tous les modèles de E,
- Pour chaque modèle M de E, vérifier que $M(f) = v$;
- Si E utilise n propositions atomiques, le nombre de modèles potentiels est 2^n.

Remarques

1. Le temps de vérification devient exponentiel avec le nombre de variables.
2. Si il existe une interprétation où $v(A) = T$, $v(B) = T$,... et $v(E) = F$ alors $A, B, \ldots \not\models E$.

Exemple

Vérifier à l'aide de la table de vérité la conséquence valide suivante :
$A \supset B, B \supset C \models A \supset C$.

A	B	C	$A \supset B$	$B \supset C$	$B \supset C$
T	T	T	**T**	**T**	**T**
T	T	F	T	F	F
T	F	T	F	T	T
T	F	F	F	T	F
F	T	T	**T**	**T**	**T**
F	T	F	T	F	T
F	F	T	**T**	**T**	**T**
F	F	F	**T**	**T**	**T**

TABLE 2.2: Exemple de la conséquence valide

Donc $A \supset C$ est une conséquence valide de $A \supset B$ et $B \supset C$.

2.3.5 Formes normales (Normalisation)

Toute formule peut se réécrire sous la forme :

$$C_1 \wedge C_2 \wedge \ldots \wedge C_n$$

Où chaque C_i (appelé clause) est elle-même de la forme :

$$p_1 \vee p_2 \vee \ldots \vee p_m \vee \neg q_1 \vee \neg q_2 \vee \ldots \vee \neg q_r$$

Où les p_j et q_k sont des atomes.
Cette forme est appelée forme normale conjonctive. Il y a un algorithme pour mettre en FNC.
La FNC est nécessaire pour pouvoir appliquer certains algorithmes.

Exemple

Transformer en forme normale conjonctive la formule donnée ci-dessous.

$$((b \vee c) \Longrightarrow a \vee d$$

1. Appliquer l'équivalence double-implique et implique-ou pour supprimer les $\Longrightarrow$ et $\Longleftrightarrow$. La formule devient donc : $(\neg(b \vee c) \vee a) \vee d$.
2. Appliquer De Morgan pour "descendre" les négations près des variables. La formule devient : $((\neg b \wedge \neg c) \vee a) \vee d$.
3. Appliquer la distributivité pour descendre les $\vee$ et remonter les $\wedge$, donc $((\neg b \vee a) \wedge (\neg c \vee a)) \vee d \equiv ((\neg b \vee a) \vee d) \wedge ((\neg c \vee a)) \vee d)$.
4. Appliquer l'associativité des $\vee$ et $\wedge \equiv (\neg b \vee a \vee d) \wedge (\neg c \vee a \vee d)$.

Il existe aussi une forme normale disjonctive ϕ^* :

$$D_1 \vee D_2 \vee ... \vee D_m$$

Où chaque D_i est lui-même de la forme :

$$r_1 \wedge r_2 \wedge ... \wedge r_u \wedge \neg s_1 \wedge \neg s_2 \wedge ... \wedge \neg s_v$$

Où les r_j et s_k sont des atomes.

2.4 Théorie de la preuve (démonstration)

Dans le chapitre précédent, nous avons vu la méthode des tables de vérité pour établir la validité et la consistance d'une formule. Cependant, cette méthode réclame, pour une formule contenant n atomes distincts, le calcul d'une table comprenant 2^n lignes. Il faut donc essayer de trouver des méthodes plus efficaces. En conséquence une autre approche, dite déductive, basée sur la formalisation de la théorie est proposée.

L'approche déductive a pour objet de calculer les conséquences logiques par l'application des règles d'inférence. Pour cela on construit des systèmes formels d'inférence composés d'axiomes (formules) et de règles d'inférence.
Il existe plusieurs systèmes d'inférence de ce type pour la logique des propositions : Système de Hilbert, système de déduction naturelle, principe de résolution de Robinson, etc.
Nous verrons dans ce chapitre le système de déduction naturelle ainsi que le principe de résolution de Robinson.

Pour spécifier notre système du calcul propositionnel formel nous avons besoin d'avoir :

1. Un alphabet (l'ensemble des symboles utilisés),
2. Un ensemble de suites finies de ces symboles qui sont appelées formules bien formées. Ce sont les phrases du langage formel,
3. Un ensemble fini de formules appelées axiomes,
4. Un ensemble fini de règles : les règles de déduction (règles d'inférence).

Définition

Le système formel du calcul propositionnel (CPF) est défini par :

1. L'alphabet est composé de :
   (a) $a, b, ..., z$ variables propositionnelles,
   (b) $\neg, \rightarrow$ Symboles primitifs,
   (c) (,) parenthèses.
2. L'ensemble des formules bien formées est défini par induction comme suit :
   (a) Chaque variable propositionnelle est une formule bien formée,
   (b) Si A et B sont des formules bien formées alors, $A \rightarrow B$ et $\neg B$ sont des formules bien formées,
   (c) L'ensemble des formules bien formées est engendré par les clauses a) et b) ci-dessus.

3. Axiomes : Pour toutes formules bien formées (fbfs) A, B et C les formules suivantes sont des axiomes du système formel CPF.
   (a) $(A \to (B \to A))$... (axiome 1),
   (b) $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$...(axiome 2),
   (c) $(\neg B \to \neg A) \to (A \to B)$...(axiome 3).
4. Règles d'inférence appelée modus ponens (MP)$(A \to B, A) \to B$ ou $\frac{A, A \to B}{B}$. En d'autres termes cette règle signifie : B est une conséquence directe de A et $(A \to B)$ où A et B sont des formules bien formées (fbfs).
5. Règle appelée modus tollens (MT) : $(A \to B, \neg B) \to \neg A$ ou $\frac{A \to B, \neg B}{\neg A}$
6. Règle de substitution (Sub) : $\frac{B}{[A=p]B}$
   La règle de substitution signifie que à partir d'une formule B contenant une variable p, on peut déduire la formule $[A/p]\ B$ obtenu en remplaçant toutes les occurrences de p dans B par une formule A.

Remarques

— Les deux symboles $\neg$ et $\to$ ont été introduits sans aucune définition. Ici ils n'ont aucun sens précis. Ce sont de simples marques (des symboles primitifs) intervenant dans la construction des formules.

— La règle Modus Ponens est la seule règle du système qui nous permet d'obtenir, à partir des axiomes, d'autres formules appelées conclusion de déduction.

Important

Un théorème est une affirmation (mathématique ou logique) qui peut être démontrée, c'est-à-dire une assertion qui peut être établie comme vraie au travers d'un raisonnement logique construit à partir des axiomes.

2.4.1 Système de déduction naturelle

Dans ce système, il n'y a pas d'axiomes, uniquement des règles d'inférence. On notera : $\frac{f1, f2, \ldots fn}{g}$

Le fait qu'on peut produire g à partir de $f_1, ..., f_n$ (inférence directe). On note :

$f \rightarrow \ldots \rightarrow g$ représente qu'on peut produire g à partir de f par une succession d'inférences. Dans tout système d'inférence on peut utiliser toutes les équivalences connues pour produire de nouvelles formules. Si on a $f \equiv g$, on a forcément $f \models g$.

Exemple

À partir de la formule suivante $\neg(p \vee q)$ nous pouvons déduire $\neg p \wedge \neg q$.

2.4.1.1 Règles d'inférence

2.4.1.2 Complétude et consistance

Idéalement un système formel doit satisfaire les deux critères de consistance et de complétude. Ces deux notions font le lien entre syntaxe et sémantique.

On se rappelle qu'un théorème est une formule qu'on peut produire par application des règles à partir des axiomes. La «*théorèmité*» d'une formule est donc une notion syntaxique. Par contre la notion de conséquence logique est sémantique, elle est définie en dehors de tout système d'inférence formel. C'est cette notion qui nous servira à vérifier la consistance et la complétude d'un système d'inférence.

Définition

Un système d'inférence est consistant si tous les théorèmes sont des conséquences logiques des formules des axiomes (tout ce qui est démontrable est «vrai »). Ou de manière équivalence, si on peut produire g à partir de l'ensemble de formules F, noté $F \vdash g$, alors g est une conséquence de F ($F \models g$).
Un système d'inférence est complet s'il permet de produire toutes les conséquences logiques des axiomes, c'est-à-dire si toute conséquence des axiomes est un théorème (tout ce qui est vrai est démontrable). Ou, de manière équivalente, $F \models g$ entraîne $F \vdash g$. Le système d'inférence naturel est consistant et complet.

2.4.2 Principe de résolution de Robinson

Le principe de résolution est formé d'une unique règle d'inférence. Sa grande simplicité de mise en œuvre en fait une méthode très utilisée.

Théorème

Toute formule est équivalente à une conjonction de clauses. On notera $C(F)$ un ensemble de clauses dont la conjonction est équivalente à F.

Exemple

$$C(p \longleftrightarrow q) = \neg p \vee q, p \vee \neg q$$

Une interprétation satisfait F si seulement si elle satisfait $C(F)$. En particulier $C(F)$ est contradictoire si seulement si F est une antilogie.

2.4.2.1 Règle de résolution

Si u est un littéral d'une clause C, on notera C/u la clause dont tous les littéraux sont ceux de C sauf u.

Exemples

1. Soit la formule suivante : $(p \vee q \vee r)\ q = p \vee r$
   La clause C est une résolvante des clauses C_1 et C_2 si il existe un littéral u tel que :

   — U est un littéral de C_1,
   — $\neg u$ est un littéral de C_2,
   — $C = (C_1/u) \vee (C_2/\neg u)$.
2. $p \vee r$ et $q \vee \neg r$ donnent la résolvante $p \vee q$,
3. $p \vee q \vee r$ et $\neg r \vee s \vee \neg t \vee q$ donnent $p \vee q \vee s \vee \neg t$,
4. $p \vee q \vee r$ et $\neg p \vee \neg q \vee s$ donnent $q \vee r \vee \neg q \vee s$ et $p \vee r \vee \neg p \vee s$, **attention elles ne donnent pas** $r \vee s$.
5. $\neg p \vee q$ et p donnent q,
6. $\neg p \vee p$ donnent la clause vide notée $\square$.

Une preuve par résolution de la clause F à partir de l'ensemble de clauses C est une suite finie de clauses $C_1, C_2, \ldots C_n$ telles que $C_i \in C$ ou bien C_i est une résolvante

de deux C_j précédents, et $Cn = F$.
Notation : $C \vdash F$.

Exemple

$p, \neg p \vee q, \neg q \vee r \vdash r$ en prenant :

- $C_1 = p, C_2 = \neg p \vee q$, dans l'ensemble de départ,
- $C_3 = q$ est une résolvante de C_1 et C_2.
- $C_4 = \neg q \vee r$, dans l'ensemble de départ,
- $C_2 = r$ est une résolvante de C_3 et C_4.

Une réfutation d'un ensemble de clauses C est une preuve de la clause vide à partir de C.
Notation : $C \vdash \square$.

Exemple de réfutation

De l'ensemble de clauses :

1. $\neg p \vee \neg q \vee r$,
2. $\neg p \vee q$,
3. P,
4. $\neg r$. On obtient successivement :
5. $\neg p \vee \neg q$ (1+4)
6. Q (2+3)
7. $\neg p$ (5+6)
8. $\square$ (3+7)

2.4.2.2 Validité de la règle de résolution

Propriété

La règle de résolution est saine (Sound), c'est-à-dire toute résolvante est une conséquence sémantique des clauses dont elle est issue.

Corollaire 1

Toute clause F déduite d'un ensemble C de clauses par résolution est une conséquence sémantique de C si $C \vdash F$ alors $C \models F$.

Corollaire 2

Un ensemble de clauses admettant une réfutation si $C \vdash \square$ alors $C \models F$.

2.4.2.3 Complétude de la méthode de résolution

Théorème

La méthode de résolution est complète pour la réfutation, c'est-à-dire, si C est un ensemble de clauses contradictoire, alors il existe une réfutation de C.

Remarques

- Elle n'est pas complète pour la déduction.
- La méthode est non déterministe, le théorème dit que la réfutation existe, il ne dit pas comment la trouver, ni si la recherche se termine (voir la figure 2.1).

$$\begin{array}{ccc} C \vdash F & \Rightarrow & C \models F \\ \Downarrow & & \Updownarrow \\ C \cup C(\neg F)\square & \Leftrightarrow & C \cup C(\neg F) \models f \end{array}$$

FIGURE 2.1: Complétude de la méthode de résolution

2.4.2.4 Quelques stratégies

Résolution positive

On n'autorise la résolution qu'entre deux clauses dont l'une est positive (n'a aucun littéral négatif).

Remarque

Il y a au moins une clause positive dans un ensemble contradictoire. (Sinon l'ensemble serait satisfait par l'interprétation toujours fausse).

Propriété

La résolution positive est complète pour la réfutation.

Exemple

de l'ensemble des clauses :

1. $\neg p \vee \neg q \vee r$
2. $\neg p \vee q$
3. p
4. $\neg r$

   On obtient successivement
5. $\neg q \vee r$ *(1 + 3)*
6. q *(2 + 3)*
7. r *(5 + 6)*
8. $\square$ *(4 + 7)*

Résolution négative

On n'autorise la résolution qu'entre deux clauses dont l'une est négative (n'a aucun littéral positif).

Remarque

Il y a au moins une clause négative dans un ensemble contradictoire. (Sinon l'ensemble serait satisfait par l'interprétation toujours vraie).

Propriété

La résolution négative est complète pour la réfutation.

Exemple

de l'ensemble des clauses :

1. $\neg p \vee \neg q \vee r$
2. $\neg p \vee q$
3. p
4. $\neg r$

   On obtient successivement

5. $\neg p \vee \neg q$ *(1 + 4)*
6. $\neg p$ *(2 + 5)*
7. $\square$ *(3 + 6)*

Résolution linéaire et résolution linéaire par entrée

Une preuve par résolution linéaire d'une clause F à partir d'un ensemble C de clauses est une suite $C_0, C_1, ...C_n$ telle que $C_0 \in C$ et pour tout i, C_i est une résolvante de C_{i-1} et d'une clause de C ou d'un C_j précédent.
La preuve est linéaire par entrée si on prend toujours une clause C (clause d'entrée).

Exemple

de l'ensemble des clauses :

1. $\neg p \vee \neg q \vee r$
2. $\neg p \vee q$
3. p
4. $\neg r$

   On obtient successivement

5. q *(2 + 3)*
6. $\neg p \vee r$ *(5 + 1)*

7. *r (6 + 3)*
8. □ *(7 + 4)*

Propriété

La résolution linéaire est complète pour la réfutation. La résolution linéaire par entrée n'est pas complète. La méthode de résolution est, dans le cas général, inefficace. Il existe cependant des cas particuliers où cette méthode est rapide.

Nous allons nous intéresser dans ce qui suit à un type particulier de forme normale, les formes normales composées de clause de Horn.

Propriété

La résolution linéaire par entrée est complète pour la réfutation des clauses de Horn.

Définition de la clause de Horn

Une clause de Horn est une clause comportant au plus un littéral positif. Il existe donc trois types de clauses de Horn :

— Celles comportant un littéral positif et au moins un littéral négatif, appelées clauses de Horn strictes ;
— Celles comportant un littéral positif et aucun littéral négatif, appelées clauses de Horn positives ;
— Celles ne comportant que des littéraux négatifs, appelées clauses de Horn négatives (et dont fait partie la clause vide ϕ).

Algorithme de résolution adapté aux clauses de Horn

L'algorithme présenté ci-dessous constitue une amélioration de l'algorithme de résolution appliqué à une forme normale N ne contenant que des clauses de Horn.

1. Si ϕ est dans N, l'ensemble est inconsistant et la résolution est terminée ;
2. Sinon, choisir une clause C et une clause P telles que P soit une clause de Horn positive réduite à p et C une clause contenant $\neg p$;

3. Calculer la résolvante R de P et C ;
4. Reprendre l'algorithme en remplaçant N par $(N \setminus \{C\}) \bigcup R$.

Cet algorithme peut se terminer de deux façons :

1. Nous avons abouti à $\emptyset$ et l'ensemble N est alors inconsistant ;
2. Il est impossible de poursuivre, car nous ne parvenons plus à trouver P et C vérifiant les conditions requises ; la formule est alors consistante et le modèle composé de chacune des clauses positives restantes (ou plus exactement le modèle composé de chacun des atomes de ces clauses), lorsque l'algorithme se termine, est un modèle de N (on appelle d'ailleurs ce modèle le modèle canonique de N).

Exemple

Nous allons utiliser cette méthode pour démontrer l'inconsistance de :

$$\{\neg p \vee r, \neg r \vee s, p, \neg s\}$$

1. Par sélection de p et $\neg p \vee r$, l'ensemble se réduit à : $\{r, \neg r \vee s, p, \neg s\}$.
2. Par sélection de r et $\neg r \vee s$, nous obtenons : $\{r, s, p, \neg s\}$.
3. Enfin, par sélection de s et $\neg s$: $\{r, s, p, \emptyset\}$.

L'ensemble de clauses est bien inconsistant.

Utilisation des clauses de Horn

— Les clauses de Horn positives, par exemple p, sont appelées faits. Il s'agit en effet de l'énoncé de la vérité logique d'un atome.

— Les clauses de Horn strictes $q \vee \neg p_1 \vee ... \vee \neg p_n$ sont équivalentes à $p_1 ..., p_n \models q$ et représentent des règles du type *si... alors...*. Elles permettent de déduire de nouveaux faits à partir de faits existants.

— Les clauses négatives peuvent se comprendre comme des buts à atteindre. Considérons que nous souhaitions prouver $\{H_1, ..., H_n\} \models (p \wedge q \wedge r)$. La partie $(p \wedge q \wedge r)$ est le but de notre résolution. En appliquant une technique d'inconsistance nous sommes ramenés à $\{H_1, ..., H_n, (\neg p \vee \neg q \vee \neg r)\} \models \emptyset$. La clause $(\neg p \vee \neg q \vee \neg r)$ est une clause de Horn négative qui modélise donc bien le but à atteindre.

Exemple

Soit $F = (p \to (q \to r)) \to ((p \to q) \to (p \to r))$ alors
$\neg F \equiv (\neg p \vee \neg q \vee r) \wedge (\neg p \vee q) \wedge p \wedge \neg r$
$C(\neg F) = \{\neg p \vee \neg q \vee r, \neg p \vee q, p, \neg r\}$
$C(\neg F) \vdash \square$
Donc F est une tautologie

2.5 Conclusion

Dans ce chapitre, nous avons présenté la logique propositionnelle définie autant que logique sans quantificateurs qui s'intéresse uniquement aux opérations logiques : la négation, la conjonction, la disjonction, l'implication et l'équivalence. Elle permet de construire des raisonnements à partir de ces connecteurs. Nous aborderons dans le prochain chapitre la logique descriptive ou souvent appelée «*logique d'ordre 1*».

Chapitre 3

Logique des prédicats (ordre 1)

3.1 Introduction

Le calcul des prédicats est considéré comme une extension du calcul propositionnel qui donne la possibilité d'introduire en même temps que les variables propositionnelles d'autres variables appartenant à un domaine arbitraire (ensemble d'entiers, de réels ou d'objets quelconques). Cette extension est obtenue grâce à l'introduction des deux quantificateurs $\forall$ et $\exists$.

L'objectif de ce chapitre est de définir la logique du premier ordre. Nous parlerons d'abord de la syntaxe, c'est à-dire comment écrire les formules, puis de leur sémantique.

3.2 Définitions

3.2.1 Prédicat

Un prédicat est une propriété ou relation qui porte sur un ou plusieurs éléments d'un domaine D. C'est une fonction de D dans T, F.
Prenons le problème suivant :

— Tout homme est mortel.
— Socrate est un homme.

— Donc, Socrate est mortel.

Nous avons déjà traduit des énoncés en logique des propositions. Supposons la traduction suivante :

— Tout homme est mortel est traduit par la proposition a.

— Socrate est un homme est traduit par la proposition b.

— Donc, Socrate est mortel est traduit par la proposition c.

Le problème s'écrit alors, en logique des propositions, $a \wedge b \rightarrow c$. Cette traduction est correcte. Mais elle est de piètre qualité.

Ce problème peut être traduit de manière adéquate en logique des prédicats :

— Hypothèse 1 : Quelque soit x appartenant au domaine D, si x est un homme, alors x est mortel $(\forall x(H(x) \rightarrow M(x)))$;

— Hypothèse 2 : Socrate à la propriété d'être un homme $(H(Socrate))$;

— Conclusion : Socrate à la propriété d'être mortel $(M(Socrate))$.

Dans cet exemple, nous avons utilisé deux prédicats (H et M), une constante (Socrate), une variable (x) et le quantificateur universel ($\forall$). Comme nous l'expliquons plus tard, ce raisonnement, ainsi formalisé en logique des prédicats, sera valide.

3.2.2 Formule

Avant de définir le langage du calcul des prédicats, l'ensemble des formules bien formées (correctement écrites) nous allons d'abord préciser :

1. **L'alphabet** du langage du calcul des prédicats est composé de :
   — $x1, x2$: les variables
   — $a1, a2$: les constantes
   — $P1, P2$: les prédicats
   — $f1, f2$: les fonctions
   — $\neg, \rightarrow$: les connecteurs
   — $\forall, \exists$: les quantificateurs
2. **Les termes** du calcul des prédicats ($CP.$) sont définis comme suit :
   (a) Les variables et les constantes sont des termes.
   (b) Si fi est une fonction à n arguments du $CP.$ et $t_1, t_2, \ldots t_n$ sont des termes alors $f_i(t_1, t_2, \ldots t_n)$ est un terme du $CP.$
   (c) L'ensemble des termes est engendré par les clauses (a) et (b).

3.2.3 Formule atomique

Une formule atomique du $CP.$ est définie par : Si P_i est un prédicat, une fonction définie d'un domaine D vers l'ensemble $\{0;1\}$ et $t_1, t_2, \ldots t_n$ sont des termes alors $P_i(t_1; t_2; \ldots t_n)$ est une formule atomique.

3.2.4 Formule bien formée

Les formules bien formées (ou formule tout simplement) sont définies par :

1. Chaque formule atomique est une formule bien formée.
2. Si A, B sont des formules bien formées du $CP.$ alors $\neg A$, $(A \to B)$, $(\forall x)A$, et $(\exists x)\ A$ sont des formules bien formées.
3. L'ensemble de toutes les formules bien formées est engendré par les clauses (1) et (2).

Exemple

Les formules suivantes sont des formules bien formées :

— $F \equiv \forall x A(x) \to (\forall y((P(f(x);0) \to (P(g(y);b)))$

— $G \equiv (P(f(a);b)) \to \forall x((\neg P(x;0)) \wedge P(f(x);g(x;f(x))))$

Parenthèsage

De la même manière qu'en logique des propositions, nous accepterons les formules dont le parenthèsage est partiellement ou complètement implicite. Pour cela, les connecteurs et les quantificateurs sont traditionnellement classés de la façon suivante (par priorité décroissante des connecteurs et quantificateurs) : $\forall, \exists, \neg, \wedge, \vee, \to, \leftrightarrow$. Dans le cas où deux connecteurs ont même priorité, et en l'absence de parenthèse, l'associativité se fait de gauche à droite. Ainsi, $\neg\forall x A \vee B$ doit se lire $((\neg(\forall x A)) \vee B)$.

3.2.5 Variables liées, variables libres

Dans la formule $\forall x A$ la formule A est dite champ du quantificateur $\forall$. La variable x est dite variable quantifiée par le quantificateur universel $\forall$. Les positions occupées

par la variable x dans la formule A sont appelées occurrences de x. Dans la formule précédente F la variable x possède deux occurrences.

Définition 1

Une occurrence d'une variable x dans une formule est dite liée si elle possède une occurrence dans le champ d'un quantificateur $\forall$ (ou $\exists$) dans cette formule. Si une occurrence d'une variable n'est pas liée, elle est libre.

Exemples

1. Soit $F \equiv \forall x A(x) \rightarrow (\forall y((P(f(x), 0) \rightarrow P(g(y), b)))$. La première occurrence de la variable x est liée par le quantificateur $\forall x$, mais la deuxième occurrence de la variable x est libre.
2. $\forall x, \exists y P(x, y) \supset Q(z, y), \forall x, \exists y P(x, y) \supset Q(z, y)$ la portée de $\exists y$, $x, \exists y P(x, y) \supset Q(z, y)$ la portée de $\forall x$. x est liée, y est liée, z est libre. (Une variable est liée à un quantificateur si elle se trouve dans sa portée).

Définition 2

Soit A une formule du calcul des prédicats. Un terme t est libre pour la variable x dans A si aucune occurrence libre de x dans A n'appartient à un champ d'un quantificateur ($\forall y$) où y est une variable dans t.

Exemple

Le terme $f(x, y)$ n'est pas libre pour y dans la formule : $\forall x P(x, y) \rightarrow \forall z Q(z, x)$.

Remarques

— Un terme t est dit clos s'il ne contient pas de variables.
— Une formule A est dite close si elle ne contient pas de variables libres.

3.3 Théorie des modèles

Comme en calcul propositionnel, nous allons nous efforcer de construire un modèle permettant de dégager une interprétation sémantique de nos formules.
Cependant, en calcul des prédicats, il n'est pas possible d'appliquer une méthode des tables de vérité directement dérivée du calcul propositionnel, en raison des domaines de valeur des variables de chacun des prédicats. En fait, pour une formule atomique $p(x_1, ..., x_n)$, nous allons avoir besoin d'une fonction (appelée fonction d'interprétation) chargée de donner un sens au symbole p, et donc de calculer sa valeur de vérité selon la valeur des $x_1, ..., x_n$.

3.3.1 Interprétation

On attribue un *sens* (une valeur de vérité) à chacune des formules en interprétant les différents symboles (fonctions, prédicats), les constantes et les variables.

Exemple

Soient $x, y \in N$, $\forall x \exists y (x \preceq y)$. Dans ce cas, on peut voir si cette formule est une proposition vraie ou une proposition fausse.

3.3.1.1 Interprétation des termes

Soit une formule F et I une interprétation de cette formule. On peut étendre l'interprétation I aux termes de F :

- à chaque symbole de constante, on associe sa valeur selon I ;
- à chaque variable, on associe la variable elle-même ;
- à chaque terme $f(t_1, ..., t_n)$, on associe le terme $f'(t'_1, ..., t'_n)$ où $t'_1, ..., t'_n$ sont les interprétations des $t_1, ..., t_n$ et f' est l'interprétation de f.

3.3.1.2 Interprétation des formules

Elle est définie par :

- Si F est un atome $p(t_1, ..., t_n)$, $I(F)$ est la fonction $p\prime(t\prime_1, ..., t\prime_n)$ où $p\prime$ est l'interprétation de p et où chaque $t\prime_i$ est l'interprétation de t_i ;

— Si F est de la forme $(\neg G)$, $(G \rightarrow H)$, $(G \leftrightarrow H)$, $(G \wedge H)$, $(G \vee H)$, $I(F)$ est définie par les mêmes lois fonctionnelles que celles définies pour le calcul propositionnel ;
— Si F est de la forme $\forall x G(x, y_1, ..., y_n)$, pour tout n-uplet $(a_1, ..., a_n) \in D_i$, $I(F)(a_1, ..., a_n) = T$ si pour toute valeur $a \in D$, $I(G)(a, a_1, ..., a_n) = T$, $I(F)(a_1, ..., a_n) = F$ sinon ;
— si F est de la forme $\exists x G(x, y_1, ..., y_n)$, pour tout n-uplet $(a_1, ..., a_n) \in D_i$, $I(F)(a_1, ..., a_n) = T$ s'il existe une valeur $a \in D$, $I(G)(a, a_1, ..., a_n) = T$, $I(F)(a, a_1, ..., a_n) = F$ sinon ;

Définitions

— Une formule A est vraie dans une interprétation I si $v(A) = T$ dans I.
— Une formule A est dite satisfaite s'il existe une interprétation I telle que A est vraie dans I.
— Une formule du calcul des prédicats est dite valide si elle est vraie pour toute interprétation. Elle est dite non valide s'il existe au moins une interprétation pour laquelle la formule n'est pas vraie (fausse) ou possède un contre-modèle.
— Une formule du calcul des prédicats est dite insatisfaisable si elle est fausse pour toute interprétation.

Notation : $I \models A$ signifie que A est vraie dans I.

3.3.2 Validité et consistance

Définitions

— **Formule valide :** Une formule F est dite valide ssi, pour toute interprétation I, on a $I(F) = T$.
— **Formule satisfiable :** Une formule A sera dite satisfiable, ou sémantiquement consistante, s'il existe une interprétation I telle que $I(A) = T$. L'interprétation est alors un modèle de A.
— **Formule invalide :** Une formule invalide est fausse dans au moins une interprétation.

— **Formule insatisfiable :** Une formule insatisfiable, ou sémantiquement inconsistante, ou encore anti-tautologie, est une formule fausse dans toute interprétation.
— **Formule contingente :** Une formule contingente est vraie dans certaines interprétations et fausse dans d'autres.
— **Conséquence logique :** Soit une formule B et une famille de n formules A_i ; On dit que B est conséquence des A_i si pour toute interprétation I telle que $\forall A_i, I(A_i) = T$, on a aussi $I(B) = T$. On note alors $A_1, A_2, ..., A_n \models B$.

3.3.3 Équivalence des formules bien formées

3.3.3.1 Formules équivalentes

Deux formules sont équivalentes quand elles ont même valeur dans toutes interprétation (notation : $A \equiv B$). Soient $A(x)$ et $B(x)$ deux formules atomiques bien formées. Les formules équivalentes de la logique des propositions demeurent équivalentes en logique des prédicats.

— $\forall x A(x) \wedge \forall x B(x) \equiv \forall x(A(x) \wedge B(x))$,
— $\exists x A(x) \vee \exists x B(x) \equiv \exists(A(x) \vee B(x))$,
— $\neg(\forall x A(x)) \equiv \exists \neg A(x)$,
— $\neg(\exists x A(x)) \equiv \forall x \neg A(x)$.

Remarques

— $\forall x A(x) \vee \forall x B(x) \not\equiv \forall x(A(x) \vee B(x))$,
— $\exists x A(x) \wedge \exists x B(x) \not\equiv \exists x(A(x) \wedge B(x))$.

3.3.3.2 Forme prénexe

Elle consiste à **accumuler les quantificateurs au début de la formule**. L'utilité de cette forme est qu'elle permet de mettre en évidence certaines relations logiques qui ne sont pas évidentes à voir sous les formes habituelles des formules.

— Une matrice est une formule du calcul des prédicats ne contenant aucun quantificateur.
— Toute formule admet une forme prénexe équivalente.

— Une forme prénexe est une formule du calcul des prédicats de la forme $Q_1x_1...Q_nx_nM$, où Q désigne $\exists$ ou $\forall$ et M est une matrice.

Théorème

Toute formule du calcul des prédicats peut être transformée en une formule équivalente sous forme normale prénexe conjonctive.

Définition

Une formule est dite sous forme normale prénexe conjonctive si elle est de la forme :

$$(Qv_1)(Qv_2)...(Qv_n)[A_{11} \vee A_{12}... \vee A_{1n}] \vee ... \vee [A_{m1} \wedge A_{m2} \vee ... \vee A_{mq}]$$

Où Q est soit le quantificateur $\forall$ ou soit le quantificateur $\exists$, les v_i sont des variables distinctes qui ont une occurrence libre dans les A_{ij}. Chaque A_{ij} est une formule atomique ou une négation d'une formule atomique.

Algorithme de construction

Pour construire la forme prénexe, il faut :

1. Supprimer les connecteurs d'équivalence et d'implication.
2. Renommer certaines variables liées de manière à n'avoir plus de variable quantifiée deux fois.
3. Supprimer les quantificateurs inutiles (dont la variable quantifiée n'apparaît pas dans leur portée) s'il y en a.
4. Transférer toutes les occurrences de la négation devant les atomes en utilisant les règles de réécriture vues pour la logique des propositions et les règles supplémentaires suivantes :
   — $\neg\forall xA \equiv \exists x\neg A$;
   — $\neg\exists xA \equiv \forall x\neg A$.
5. Faire passer les quantificateurs en tête en utilisant les règles de réécriture suivantes (et en utilisant l'associativité, la commutativité ou le renommage de variable si nécessaire) :

— $(\forall x A \wedge B) = \forall x(A \wedge B)$ si B ne contient pas x ;
— $(\exists x A \wedge B) = \exists x(A \wedge B)$ si B ne contient pas x ;
— $(\forall x A \vee B) = \forall x(A \vee B)$ si B ne contient pas x ;
— $(\exists x A \vee B) = \exists x(A \vee B)$ si B ne contient pas x.

Exemple 01

Transformer la formule suivante sous forme prénexe :

$$\forall x p(x) \wedge \exists y q(y) \rightarrow \exists y(p(y) \wedge q(y))$$

1. $\neg(\forall x p(x) \wedge \exists y q(y)) \vee \exists y(p(y) \wedge q(y))$ par suppression de $\rightarrow$;
2. $\neg(\forall x p(x) \wedge \exists y q(y)) \vee \exists z(p(z) \wedge q(z))$ par renommage des variables ;
3. $(\exists x \neg p(x) \vee \forall y \neg q(y)) \vee \exists z(p(z) \wedge q(z))$ par transfert de la négation ;
4. $\exists x \forall y \exists z(\neg p(x) \vee \neg q(y) \vee (p(z) \wedge q(z)))$ par déplacement des quantificateurs.

Exemple 02

Transformer la formule suivante en forme prénexe conjonctive :

$$\forall x[(\forall y p(x) \vee \forall z q(z;y)) \rightarrow \neg \forall y r(x;y)]$$

1. $(\forall x)[(p(x) \wedge (\forall z) q(z;y)) \rightarrow \neg(\forall y) r(x;y)]$;
2. $\forall x[(p(x) \wedge (\forall z) q(z;y)) \rightarrow \neg \forall y_1 r(x;y_1)]$;
3. $\forall x[\neg(p(x) \wedge \exists z \neg q(z;y)) \vee \exists y_1 \neg r(x;y_1)]$;
4. $\forall x \exists z \exists y_1[\neg(p(x) \wedge \neg q(z;y)) \vee \neg r(x;y_1)]$;
5. $\forall x \exists z \exists y_1[\neg(p(x) \vee \neg r(x;y_1)) \wedge (\neg q(z;y) \vee \neg r(x;y_1)]$.

3.3.4 Forme de Skolem

La forme Skolem est constituée que des quantifications universelles. La procédure de construction est comme suit :

1. Prendre la fermeture existentielle de D (c.à.d quantifier les variables libres de D par le quantificateur $\exists$) ;
2. Éliminer dans D tous les quantificateurs redondants ;
3. Renommer chaque variable quantifiée dans D plus d'une fois ;
4. Éliminer les occurrences des connecteurs autres que $\neg$, $\vee$ et $\wedge$.
5. Pousser à droite tous les connecteurs en remplaçant :
   $\neg(\forall x A)$ par $\exists(\neg A)$;
   $\neg(\exists x A)$ par $\forall x(\neg A)$;
   $\neg(A \vee B)$ par $\neg A \wedge \neg B$;
   $\neg(A \wedge B)$ par $\neg A \vee \neg B$;
   $\neg(\neg A)$ par A.
   Jusqu'à ce que toutes les occurrences de $\neg$ précède immédiatement une formule atomique.
6. Pousser les quantificateurs à droite en remplaçant :
   — $\exists(A \vee B)$ par
     — $A \vee \exists B$ si x n'est pas libre dans A ;
     — $\exists A \vee B$ si x n'est pas libre dans B.
   — $\exists(A \wedge B)$ par
     — $A \wedge \exists B$ si x n'est pas libre dans A ;
     — $\exists A \wedge B$ si x n'est pas libre dans B.
   — $\forall x(A \wedge B)$ par
     — $A \wedge \forall x B$ si x n'est pas libre dans A ;
     — $\forall x A \wedge B$ si x n'est pas libre dans B.
7. Éliminer les quantificateurs existentiels. Prendre la formule la plus à gauche de la forme $\exists y B(y)$ et la remplacer par $B(f(x_1, ..., x_n))$ où
   — $x_1, ..., x_n$ sont des variables distinctes de $\exists y B(y)$ quantifiées universellement à gauche de $\exists y B(y)$.
   — f est fonction à n arguments qui n'occure pas dans la formule. Répéter ce processus jusqu'à l'élimination complète des tous les quantificateurs extensionnels. Dans le cas particulier $n = 0$, remplacer $\exists y B(y)$ par $B(a)$ où a est une nouvelle constante qui n'occure pas dans la formule.
8. Déplacer les quantificateurs $\forall$ à gauche ;
9. Distribuer $\wedge$ et $\vee$;
10. Simplifier en utilisant les règles préservant la satisfaction.

Exemple

Transformer la formule suivante sous forme Skolem.

$$\forall x(p(x) \rightarrow \exists z(\neg\forall y(q(x,y) \rightarrow p(f(t))) \wedge \forall y(q(x,y) \rightarrow p(x))))$$

1. Prendre la fermeture existentielle de D et éliminer le quantificateur redondant $\exists z$.
   $\exists t\forall x(p(x) \rightarrow (\neg\forall y(q(x,y) \rightarrow p(f(t))) \wedge \forall y(q(x,y) \rightarrow p(x))))$
2. Renommer la variable y, elle est quantifiée deux fois.
   $\exists t\forall x(p(x) \rightarrow (\neg\forall y(q(x,y) \rightarrow p(f(t))) \wedge \forall z(q(x,z) \rightarrow p(x))))$
3. Éliminer toutes les occurrences de $\rightarrow$.
   $\exists t\forall x(\neg p(x) \vee (\neg\forall y(\neg q(x,y) \vee p(f(t))) \wedge \forall z(\neg q(x,z) \vee p(x))))$
4. Déplacer $\neg$.
   $\exists t\forall x(\neg p(x) \vee (\exists y(q(x,y) \wedge \neg p(f(t))) \wedge \forall z(\neg q(x,z) \vee p(x))))$
5. Pousser les quantificateurs $\exists y$ et $\forall z$ à droite.
   $\forall x(\neg p(x) \vee ((q(x,g(x)) \wedge \neg p(f(a))) \wedge (\forall z(\neg q(x,z) \vee p(x))))$
6. Éliminer les quantificateurs $\exists t$ et $\exists y$.
   $\forall x(\neg p(x) \vee ((q(x,g(x)) \wedge \neg p(f(a))) \wedge (\forall z(\neg q(x,z) \vee p(x))))$
7. Déplacer le quantificateur $\forall$ à gauche.
   $\forall x\forall z(\neg p(x) \vee ((q(x,g(x)) \wedge \neg p(f(a))) \wedge ((\neg q(x,z) \vee p(x))))$
8. Après distribution de $\vee$ et $\wedge$ simplification.
   $\forall x\{(\neg p(x) \vee ((q(x,g(x))) \wedge \neg p(f(a))\}$

3.3.5 Forme Clausale

Un littéral est défini comme étant une formule atomique ou une négation d'une formule atomique. Une clause est une disjonction de plusieurs littéraux. Une formule du calcul des prédicats est dite sous forme clausale si elle est sous la forme :

$$\forall x_1 ... \forall x_n (C_1 \wedge C_2 \wedge ... \wedge C_k)$$

Algorithme de construction

Pour construire la forme clausale, il faut :

1. Prendre la fermeture existentielle de D (c.à.d. quantifier les variables libres de D par le quantificateur $\exists$),
2. Éliminer dans D tous les quantificateurs redondants,
3. Renommer chaque variable quantifiée dans D plus d'une fois,
4. Éliminer les occurrences des connecteurs autres que : $\neg, \vee$ et $\wedge$.
5. Pousser à droite tous les connecteurs en remplaçant :
   $\neg(\forall x A)$ par $\exists x(\neg A)$;
   $\neg(\exists x A)$ par $\forall x(\neg A)$;
   $\neg(A \vee B)$ par $\neg A \wedge \neg B$;
   $\neg(A \wedge B)$ par $\neg A \vee \neg B$;
   $\neg(\neg A)$ par A.

Exemple

Si nous partons de la formule skolémisée :

$$sk(A_2) = \forall x \forall y (P(x, f(x)) \rightarrow (Q(f(x), y) \wedge R(y, g(x, y)))$$

1. l'élimination des quantificateurs universels, donne :
   $\neg P(x, f(x)) \vee (Q(f(x), y) \wedge R(y, g(x, y)))$
2. l'application de la transformation $\rightarrow$, $\vee$, on obtient :
   $\neg P(x, f(x)) \vee (Q(f(x), y) \wedge R(y, g(x, y)))$
3. Puis par la distributivité, on obtient :
   $(\neg P(x, f(x)) \vee Q(f(x), y)) \wedge (\neg P(x, f(x)) \vee R(y, g(x, y)))$
4. Nous obtenons à la fin les deux clauses :

   (a) $\neg P(x, f(x)) \vee Q(f(x), y)$
   (b) $\neg P(x, f(x)) \vee R(y, g(x, y))$

Remarque

Toute formule est équivalente à une formule prénexe, Skolem et clausale.

3.4 Conclusion

Dans ce chapitre, nous avons présenté la logique des prédicats considérée plus riche que la logique propositionnelle par l'inclusion des quantificateurs universels au même temps que les variables propositionnelles. A travers les prochaines annexes, nous proposons une suite d'exercices corrigés.

Annexe A

Série d'exercices

Exercice 1

Parmi les assertions suivantes, lesquelles sont vraies, lesquelles sont fausses et pourquoi ?

1. Si Napoléon était chinois alors $3 - 2 = 2$;
2. Soit Cléopâtre était chinoise, soit les grenouilles aboient ;
3. Soit les roses sont des animaux, soit les chiens ont 4 pattes ;
4. Si l'homme est un quadrupède, alors il parle ;
5. Les roses ne sont ni des animaux, ni des fleurs ;
6. Alger est en Algérie ou Madrid est en chine.

Exercice 2

Soient (P), (Q) et (R) trois propositions, donner la négation de

1. (P) et $(non(Q)$ ou $(R))$,
2. $((P)et(Q)) \rightarrow (R)$

Exercice 3

Démontrer les règles suivantes à l'aide d'une table de vérité.

1. $(A.B)(A.B).C = (B.C)A.(B.C)$,
2. $\overline{A+B} = \overline{A}.\overline{B}$,
3. $\overline{A.B} = \overline{A} + \overline{B}$

Exercice 4

Écrire l'expression $\overline{A + \overline{B}.C}$ sous forme normale conjonctive et puis sous forme normale disjonctive.

Exercice 5

Soient les quatre assertions suivantes :

1. $\exists x \in R \forall y \in R x + y > 0$;
2. $\forall x \in R \exists y \in R x + y > 0$;
3. $\forall x \in R \forall y \in R x + y > 0$;
4. $\exists x \in R \forall y \in R y^2 > x$.

Les assertions 1, 2, 3, 4 sont-elles vraies ou fausses ? Donner leur négation.

Exercice 6

Traduisez les énoncés suivants en formule logique :

1. Quand il fait beau, Jean est heureux ; il fait soleil ; donc, Jean est heureux.
2. Quand il fait beau, Jean est heureux ; or, il fait mauvais ; donc Jean est malheureux.
3. Quand il fait beau, Jean est heureux ; or, Jean est malheureux ; donc il fait mauvais.

Exercice 7

Les « formules » suivantes (voir la table A.1) sont-elles des formules bien formées ?

$a \vee b \wedge c$	$\neg a \vee \neg b \wedge c$	$a \vee \neg b \wedge c$	$a\neg \vee b \wedge c$
(a)	$(a)b$	$\neg b(a)$	$a \vee \neg(b \wedge c)$
$a\neg(\vee b \wedge c)$	$a \rightarrow b$	$a \leftarrow a$	$a \rightarrow b \leftrightarrow c$
$a \rightarrow \neg(b \leftrightarrow c)$	$a\neg \rightarrow b$	$a\vee(b \leftrightarrow c)\neg \rightarrow d$	$a \vee (b \leftrightarrow c) \rightarrow \neg c \rightarrow d$

Table A.1: Table des formules

Exercice 8

Explicitez le parenthèsage implicite des formules suivantes :

1. $a \rightarrow b \rightarrow c$;
2. $a \vee b \wedge c$;
3. $a \vee b \wedge c \leftrightarrow d \rightarrow \neg e \vee f \wedge g$

Simplifiez au maximum le parenthèsage des formules suivantes (voir la table A.2) :

(a)	$((a \vee b))$	$((a) \wedge (b))$
$\neg(((a) \wedge b))$	$a \vee (b \wedge c)$	$(a \vee b) \wedge c$
$(a \wedge (b \rightarrow c))$	$((a \vee b) \wedge c) \leftrightarrow e$	$(((a\vee b)\wedge c) \leftrightarrow e) \rightarrow f$
$(((a \rightarrow b) \rightarrow c) \rightarrow d)$	$(a \wedge (b \wedge c))$	$(a \rightarrow (b \rightarrow c))$
$(\neg(a \vee b))$	$((a \wedge b) \rightarrow c)$	$((a\wedge b)\vee c) \leftrightarrow (e \rightarrow f)$

Table A.2: Table des formules à simplifier

Exercice 9

A l'aide de l'arbre de construction (décomposition), vérifier si les chaines de symboles suivantes sont des Fbf.

$((\neg(P \vee Q) \rightarrow \neg\neg\neg Q \leftrightarrow R)$	$((P \supset Q) \supset R) \vee S) \vee T))$
$A \supset B \supset C \equiv ((D \equiv E) \wedge A \vee B)$	$(B \supset D) \vee (A \vee C) \equiv E \wedge \wedge\neg B \vee \neg D)$

Table A.3: Table des formules à vérifier

Exercice 10

Vérifier au moyen des tables de vérité si les conséquences sémantiques sont valides :

1. $A, (A \supset B) \models B$;
2. $A, (B \supset C), (A \supset (B \supset C)) \models C$;
3. $(A \supset C), (C \supset B), \neg A \models \neg B$.

Exercice 11

Donner les formes normales conjonctives et disjonctives des propositions suivantes à l'aide de tables de vérité :

1. $(((P \wedge \neg Q) \vee (R \vee \neg P)) \supset (R \equiv \neg P))$;
2. $(((\neg P \vee R) \wedge (Q \wedge P)) \wedge (\neg R \vee Q))$.

Exercice 12

Soit le raisonnement suivant :

— Quand il fait soleil, je mets mes lunettes ou je ne sors pas.

— Je ne reste à la maison que sans lunettes et par temps gris.

Donc si je ne mets pas mes lunettes, c'est qu'il fait gris.

1. Formaliser ce raisonnement en utilisant les variables suivantes :
   s : il fait soleil, l : je mets mes lunettes, m : je reste à la maison.
2. Montrer que le raisonnement ci-dessus est correct (valide) :
   (a) en utilisant la table de vérité ;
   (b) en utilisant une mise en forme normale par le calcul (algébriquement).

Exercice 13

Considérons l'extension du langage du calcul propositionnel (CP) par les deux symboles

— Construire les tables de vérité des formules propositionnelles :

— Montrer que dans cette extension, les ensembles

Exercice 14

Montrez que les formules suivantes sont des théorèmes :

1. $(p \to (q \to r)) \to (q \to (p \to r))$;
2. $(p \to q) \to ((r \to p) \to (r \to q))$.

Exercice 15

Soient les trois formules $f1$, $f2$ et $f3$ suivantes :
$f1 \equiv (A \to (A \to B)) \to (A \to B)$;
$f2 \equiv ((A \to B) \to C) \to (B \to C)$;
$f3 \equiv ((A \to B) \to (B \to C)) \to (A \to ((A \to B) \to C))$.

1. Montrer, à l'aide du théorème de déduction, que $f1$, $f2$ et $f3$ sont des théorèmes.
2. Montrer, maintenant, que $f1$, $f2$ et $f3$ sont des théorèmes ; et cela sans utiliser d'hypothèses.

Exercice 16

Utiliser la méthode de résolution propositionnelle pour montrer si la formule C est une conséquence logique des formules A et B ;
Où : $A = (p \to (q \vee r))$; $B = \neg r$ et $C = (\neg q \to \neg p)$.

Exercice 17

1. En utilisant la méthode de résolution propositionnelle, démontrer que la formule suivante est une tautologie :
   $F = (a \wedge \neg b) \vee (\neg a \wedge \neg(b \vee c)) \vee (\neg c \wedge b) \vee (b \wedge c \wedge a) \vee (c \wedge \neg a)$.
2. En déduire ce qu'il en est de la validité de la formule A suivante :
   $A = (a \to b) \wedge (\neg a \to (b \vee c)) \wedge (\neg c \to \neg b) \wedge ((b \wedge c) \to \neg a) \wedge (c \to a)$.

Exercice 18

Formalisez les phrases suivantes dans le langage des prédicats, en conservant autant de structure que possible.

1. Pierre marche et Jean court.
2. Si Pierre court, alors Pierre sera fatigué.
3. Marie viendra, mais pas Jeanne.
4. Si le président de la République ne répond pas aux questions, alors l'éditorialiste écrira un article ravageur.
5. Si cet homme ou son ami reviennent dans le quartier, je ferai signe.

Exercice 19

Dans les formules suivantes, dites si les variables utilisées sont des variables libres ou des variables liées.

$p(x) \vee q(y)$	$\forall x p(x) \vee q(y)$
$\exists z(\forall x p(x,z) \vee q(y,z))$	$\forall x p(x) \vee q(y,z)$
$\forall x p(x) \vee q(x)$	$\exists z(p(x,z) \vee \exists y q(y,z))$
$p(x,y) \vee p(y,x) \wedge p(x,x) \rightarrow p(x,y)$	$\forall x \forall y(p(x,y) \vee p(y,x)) \wedge p(x,x) \rightarrow p(x,y)$
$\forall x \forall y(p(x,y) \vee p(y,x)) \wedge p(x,x) \rightarrow \exists x \forall y p(x,y)$	$\forall x \forall y(p(x,y) \vee p(y,x)) \wedge \forall x p(x,x) \rightarrow \exists x \forall y p(x,y)$

Table A.4: Table des variables

Exercice 20

Traduisez les phrases suivantes dans le langage des prédicats. Conservez autant de structure que possible.

1. Tous les plombiers sont des hommes ;
2. Pierre est riche ;
3. Si Pierre est un plombier, Pierre est riche ;
4. Quelques plombiers ne sont pas riches ;

5. Aucun plombier n'est riche.

Utilisez les traductions suivantes pour les prédicats :

— P_x : x est plombier ;

— H_x : x est un homme ;

— R_x : x est riche.

Exercice 21

Traduisez les formules en français. Utilisez les traductions des prédicats ci-dessous (table A.5), ainsi que :

— Q_x : x habite à Quimper ;

— a : Antoine ;

— b : Béatrice ;

— c : Christine.

P_a	Q_c	R_b
$\exists x Q_x$	$\forall x(P_x \to Q_x)$	$\exists x(Q_x \wedge \neg P_x)$

Table A.5: Table des prédicats

Éliminer les parenthèses dans ce qui suit :$\forall z(\exists y(\forall x(R(x,y) \wedge P(z))))$.

Annexe B

Corrigés des exercices

Exercice 1

1. Il s'agit, ici d'une implication. « Napoléon est chinois » est faux et « $3-2=2$ » est faux, or la seule possibilité pour qu'une implication soit fausse est qu'une assertion vraie implique une assertion fausse, donc l'assertion 1 **est vraie**.
2. Une phrase, en français, du genre « soit..., soit...» se traduit mathématiquement par «... ou... » « Cléopâtre était chinoise » est faux et « les grenouilles aboient » est faux donc l'assertion 2 **est fausse**.
3. « Les roses sont des animaux » est faux et « les chiens ont 4 pattes » est vrai, donc l'assertion 3 **est vraie**.
4. « L'homme est un quadrupède » est faux et « il parle » est vrai, donc l'assertion 4 **est vraie**.
5. « Les roses ne sont ni des animaux, ni des fleurs » peut se traduire par « les roses ne sont pas des animaux et les roses ne sont pas des fleurs ». « les roses ne sont pas des animaux » est vrai et « les roses ne sont pas des fleurs » est faux donc « les roses ne sont ni des animaux, ni des fleurs » , donc l'assertion 5 est fausse.
6. « Alger est en Algérie » est vrai et « Madrid est en chine » est faux, donc « Alger est en Algérie ou Madrid est en chine » est vraie.

Exercice 2

1. $non((P)$ et $(non(Q)$ ou $(R)))$ $\equiv$ $(non(P)$ ou $non(non(Q)$ ou $(R)))$ $\equiv$ $(non(P)$ ou $((Q)$ et $non(R))$ $\equiv$ $(non(P)$ ou $(Q))$ et $(non(P)$ ou $non(R))$ $\equiv$ $(non(P)$ ou $(Q))$ et $non((P)$ et $(R))$
   Les deux dernières équivalences logiques me paraissent acceptables, parce qu'il y a souvent différentes façon d'exprimer une négation, ensuite il faut voir dans les exercices comment se présentent les propositions (P), (Q) et (R).
2. $non(((P)$ et $(Q)) \to (R)) \equiv ((P)et(Q))$ et $non(R) \equiv (P)$ et $(Q)etnon(R)$.

Exercice 5

1. est fausse. Car sa négation qui est $\forall x \in R \exists y \in R x + y \leq 0$ est vraie. Étant donné $x \in R$ il existe toujours un $y \in R$ tel que $x + y \leq 0$, par exemple on peut prendre $y = -(x+1)$ et alors $x + y = x - x - 1$.
2. est vraie, pour un x donné, on peut prendre (par exemple) $y = -x + 1$ et alors $x + y = 1 > 0$. La négation de (2) est $\exists x \in R \forall y \in R x + y \leq 0$.
3. $\forall x \in R \forall y \in R x + y > 0$ est fausse, par exemple $x = -1$, $y = 0$. La négation est $\exists x \in R \exists y \in R x + y \leq 0$.
4. est vraie, on peut prendre $x = -1$. La négation est : $\forall x \in R \exists y \in R y^2 \leq x$.

Exercice 6

Ci-dessous une suite de propositions élémentaires qui nous aiderons à formaliser les énoncés.

1. $b \equiv$ il fait beau.
   $h \equiv$ Jean est heureux.
   Hypothèse 1 : $H_1 \equiv b \to h$ (Quand il fait beau, Jean est heureux).
   Hypothèse 2 : $H_2 \equiv b$ (il fait soleil).
   Conclusion : $C \equiv h$ (Jean est heureux).
   Formulation : $H_1 \wedge H_2 \to C \equiv (b \to h) \wedge b \to h$.
2. $b \equiv$ il fait beau.
   $h \equiv$ Jean est heureux.

Hypothèse 1 : $H_1 \equiv b \to h$ (Quand il fait beau, Jean est heureux).
Hypothèse 2 : $H_2 \equiv \neg b$ (il fait mauvais).
Conclusion : $C \equiv \neg h$ (Jean est malheureux).
Formulation : $H_1 \wedge H_2 \to C \equiv (b \to h) \wedge \neg b \to \neg h$.

3. $b \equiv$ il fait beau.
   $h \equiv$ Jean est heureux.
   Hypothèse 1 : $H_1 \equiv b \to h$ (Quand il fait beau, Jean est heureux).
   Hypothèse 2 : $H_2 \equiv \neg h$ (Jean est malheureux).
   Conclusion : $C \equiv \neg b$ (il fait mauvais).
   Formulation : $H_1 \wedge H_2 \to C \equiv (b \to h) \wedge \neg h \to \neg b$.

Exercice 8

Explicitez le parenthèsage implicite des formules suivantes :

1. $a \to b \to c \equiv ((a \to b) \to c)$
2. $a \vee b \wedge c \equiv (a \vee (b \wedge c))$
3. $a \vee b \wedge c \leftrightarrow d \to \neg e \vee f \wedge g \equiv ((a \vee (b \wedge c)) \leftrightarrow (d \to ((\neg e) \vee (f \wedge g))))$

Simplifiez au maximum le parenthèsage des formules suivantes (voir la table B.1) :

$(a) \equiv a$	$((a \vee b)) \equiv a \vee b$	$((a) \wedge (b)) \equiv a \wedge b$
$\neg(((a) \wedge b)) \equiv \neg(a \vee b)$	$a \vee (b \wedge c) \equiv \neg(a \wedge b)$	$(a \vee b) \wedge c \equiv a \vee b \wedge c$
$(a \wedge (b \to c)) \equiv a \wedge (b \to c)$	$((a \vee b) \wedge c) \leftrightarrow e \equiv (a \vee b) \wedge c \leftrightarrow e$	$(((a \vee b) \wedge c) \leftrightarrow e) \to f \equiv ((a \vee b) \wedge c \leftrightarrow e) \to f$
$(((a \to b) \to c) \to d) \equiv a \to b \to c \to d$	$(a \wedge (b \wedge c)) \equiv a \wedge b \wedge c$	$(a \to (b \to c)) \equiv a \to (b \to c)$
$(\neg(a \vee b)) \equiv \neg(a \vee b)$	$((a \wedge b) \to c) \equiv a \wedge b \to c$	$((a \wedge b) \vee c) \leftrightarrow (e \to f) \equiv a \wedge b \vee c \leftrightarrow e \to f$

Table B.1: Simplification des formules

Exercice 12

1. Avec les variables s, l et m dénotant : s : il fait soleil ; l : je mets mes lunettes ; m : je reste à la maison ; on peut formaliser le raisonnement donné avec une

formule du calcul propositionnel suivant :
$((s \to (l \vee m)) \wedge (m \to (\neg l \wedge \neg s))) \to (\neg l \to \neg s)$

2. Le raisonnement considéré est valide car la formule (A) est une tautologie ; montré à la table de vérité (Figure B.1) : soient le sous formules : $I = s \to (l \vee m)$, $II = m \to (\neg l \wedge \neg s)$, $III = (\neg l \to \neg s)$ avec $(A) = (I \wedge II) \to III$.
Par une méthode algébrique : $I = \neg S \wedge l \wedge m = (l \vee \neg s) \vee m) = (\neg l \to \neg s) \vee m$ puisque $m \to (\neg l \wedge \neg s)$ alors $I \to (\neg l \to \neg s) \wedge (\neg l \wedge \neg s)$.
Or $(\ negl \to \neg s) \vee (\neg l \wedge \neg s) = (\neg l \to \neg s)$ car $(A\ toB) \vee (A \wedge B) = (A \to B))$
On aura donc $(I \wedge II) \to (\neg l \to \neg s) \vee (\neg l \wedge \neg s) = (\neg l \to \neg s)$; c'est à dire $(I \wedge II) \to III$ est une tautologie.

s	l	m	$l \vee m$	I	$\neg l \wedge \neg s$	II	III	$I \wedge II$	**(A)**
0	0	0	0	1	1	1	1	1	1
0	0	1	1	1	1	1	1	1	1
0	1	0	1	1	0	1	1	1	1
0	1	1	1	1	0	0	1	0	1
1	0	0	0	0	0	1	0	0	1
1	0	1	1	1	0	0	0	0	1
1	1	0	1	1	0	1	1	1	1
1	1	1	1	1	0	0	1	0	1

FIGURE B.1: Table de vérité de la formule A

Exercice 13

Rappel

Un système complet de connecteurs C est un ensemble de connecteurs logiques tel que toute formule propositionnelle est équivalente à une formule n'utilisant que les connecteurs de C.

1. Table de vérité B.2
2. $\{\vee, \to\}$ est un ensemble complet de connecteurs car on peut exprimer les autres connecteurs usuels en fonction des éléments de cet ensemble :
$\neg A = A \to \bot$

A	B	T	$\perp$	$B \wedge T$	$A \rightarrow (B \wedge T)$	$B \vee \perp$	$A \rightarrow (B \vee \perp)$	$B \rightarrow \perp$	$A \rightarrow (B \rightarrow \perp)$
0	0	1	0	0	1	0	1	1	1
0	1	1	0	1	1	1	1	0	1
1	0	1	0	0	0	0	0	1	1
1	1	1	0	1	1	1	1	0	0

FIGURE B.2: Table de vérité des formules propositionnelles

$$A \wedge B = \neg(\neg A \vee \neg B) = (\neg A \vee \neg B)) \rightarrow \perp = ((A \rightarrow \perp) \vee (B \rightarrow \perp)) \rightarrow \perp$$
$$A \leftrightarrow B = (A \rightarrow B) \wedge (B \rightarrow A) = (((A \rightarrow B) \rightarrow \perp) \vee ((B \rightarrow A)\perp)) \rightarrow \perp$$

$\{\rightarrow\}$ constitue à lui seul un ensemble complet complet de connecteurs : en effet il suffit de remarquer que $A \vee B = (A \rightarrow B) \rightarrow B$.

Exercice 16

On a $A, B \models C$ si et seulement si $A, B, \neg C \models \square$.
On écrit A, B et $\neg C$ sous forme clausale :
$A \equiv \neg P \vee q \vee r$ (une clause),
$B \equiv \neg r$ (une clause),
$\neg C \equiv \neg q \wedge p$ (deux clauses).
On a donc :
$C_1 = \neg p \vee q \vee r$
$C_2 = \neg r$
$C_3 = \neg q$
$C_4 = p$
$C_5 = q \vee r$ Res(C_1, C_4)
$C_6 = r$ $Res(C_3, C_5)$
$C_7 = \square$ $Res(C_2, C_6)$
Donc C est une conséquence logique de A et B.

Exercice 17

1. Pour montrer que F est une tautologie, on démontre que $\neg F$ est une contradiction.

$F = (a \wedge \neg b) \vee (\neg a \wedge \neg(b \vee c)) \vee (\neg c \wedge b) \vee (b \wedge c \wedge a) \vee (c \wedge \neg a)$.

On a donc :

$C_1 = \neg a \vee b$

$C_2 = a \vee b \vee c$

$C_3 = c \vee \neg b$

$C_4 = \neg b \vee \neg c \vee \neg a$

$C_5 = \neg c \vee a$

A partir de cela, on obtient $\neg F$.

$C_6 = b \vee c$ $\mathrm{Res}(C_1, C_2)$

$C_7 = c$ $\mathrm{Res}(C_3, C_6)$

$C_8 = \neg c \vee \neg a$ $\mathrm{Res}(C_1, C_4)$

$C_9 = \neg c$ $\mathrm{Res}(C_5, C_8)$

$C_{10} = \square$ $\mathrm{Res}(C_7, C_9)$

Donc $\neg F \models \square$, d'où : $\models F$.

2. En effectuant le calcul, on obtient $A \equiv \neg F$; donc A est une contradiction.

Exercice 20

1. Tous les plombiers sont des hommes.

   $\forall x(P_x \Rightarrow H_x)$

2. Pierre est riche.

   a : Pierre, R_a

3. Si Pierre est un plombier, Pierre est riche.

   a : Pierre, $P_a \Rightarrow R_a$

4. Quelques plombiers sont riches.

   $\exists(P_x \wedge R_x)$

5. Aucun plombier n'est riche.

   $\neg\exists(P_x \wedge R_x)$

Annexe C

Examen corrigé

Questions de cours

1. Dire si l'affirmation suivante est correcte ou non, justifier votre réponse.
   — Soit **valid** un algorithme qui répond vrai pour une formule P si la formule P est valide et faux sinon. A partir de **valid**, il est possible de construire un algorithme **sat** qui répond vrai si son entrée est une formule satisfiable et faux sinon.
2. Écrire à l'aide de quantificateurs les propositions suivantes :
   — Le carré de tout réel est positif.
   — Certains réels sont strictement supérieurs à leur carré.

Correction

1. Vrai : en effet une formule A est insatisfiable si et seulement si la formule $\neg A$ est valide donc on teste si $\neg A$ est valide auquel cas auquel cas A n'est pas satisfiable sinon A est satisfiable.
2. — $\forall x \in R, x^2 \geq 0$;
   — $\exists x \in R / x \geq x^2$.

Exercice 1

Trois collègues, Albert, Bernard et Charles déjeunent ensemble chaque jour ouvrable. Les affirmations suivantes sont vraies :

1. Si Albert commande un dessert, Bernard en commande un aussi.
2. Chaque jour, soit Bernard, soit Charles, mais pas les deux, commandent un dessert.
3. Albert ou Charles, ou les deux, commandent chaque jour un dessert.
4. Si Charles commande un dessert, Albert fait de même.

Question :*Exprimer les données du problème comme des formules propositionnelles.*

Correction

On introduit des variables propositionnelles a, b, c qui représentent le fait que Albert (a), Bernard (b) et Charles (c) prennent un dessert. On traduit ainsi le problème :

1. $a \to b$;
2. $(b \wedge \neg c) \vee (\neg b \vee c)$ ou $(b \vee c) \wedge (\neg(b \wedge c))$;
3. $a \vee c$;
4. $c \to a$.

Exercice 2

On définit le connecteur de Sheffer noté | (barre de Sheffer, ou encore NAND) par :$p|q = \neg(p \wedge q)$.

1. Donner la table de vérité de la formule $(p|q)$.
2. Donner la table de vérité de la formule $((p|q)|(p|q))$.
3. On veut maintenant exprimer les connecteurs usuels en utilisant la barre de Sheffer, et rien qu'elle.
   (a) Donner la table de vérité de la formule $(p|p)$ et en déduire que le connecteur $\neg$ peut être défini en n'utilisant que la barre de Sheffer.

(b) Trouver une formule équivalente à $p \vee q$, qui n'utilise que la barre de Sheffer (éventuellement plusieurs fois).

(c) Trouver une formule équivalente à $p \to q$, qui n'utilise que la barre de Sheffer (éventuellement plusieurs fois).

4. D'après les questions précédentes quel système complet peut-on former ? Expliquer.

Correction

1. Voir la table C.1.

p	q	$p\|q$
V	V	F
V	F	V
F	V	V
F	V	V

TABLE C.1: Table de vérité $(p|q)$

2. On retrouve la table C.2 de vérité de $p \wedge q$.

p	q	$(p\|q)\|(p\|q)$
V	V	V
V	F	F
F	V	F
F	V	F

TABLE C.2: Table de vérité $(p|q)|(p|q)$

3. (a) On peut poser $\neg p \equiv (p|p)$

p	$(p\|p)$
V	F
F	V

TABLE C.3: Table de vérité $(p|p)$

(b) On a $p \vee q \equiv (\neg p \wedge \neg q) \equiv (\neg p|\neg q)$ on en déduit un codage de $p \vee q$ est : $(p|p)|(q|q)$.

(c) On a $p \to q \equiv \neg(p \wedge \neg q) \equiv p|\neg q$ et donc en déduit un codage de $p \to q$ qui est : $p|(q|q)$.

Exercice 3

Soit la formule P définie comme $(p \to (q \to r)) \to (r \vee \neg p)$

1. Donner la table de vérité de la formule P ;
2. Dire si la formule est valide, satisfiable, insatisfiable ?
3. La formule P a-t-elle un modèle ? Si oui lequel ?
4. Donner la FNC et la FND de la formule P.

Correction

1. Table de vérité :

p	q	r	$q \to r$	$p \to (q \to r)$	$(p \to (q \to r)) \to (r \vee \neg p)$
		V			V
F		F			V
V	V	F	F	F	V
V	F	F	V	V	F

Table C.4: Table de vérité de P

2. La formule n'est pas valide (une ligne de la table où elle est fausse). Elle est satisfiable (une ligne où elle est vraie) et donc elle n'est pas insatisfiable.
3. La formule a plusieurs modèles, par exemple $\{p \to V; q \to V; r \to V\}$.
4. Forme normale conjonctive qui exprime que l'on n'est pas sur la ligne où la valeur de la formule est fausse : $\neg p \vee q \vee r$. C'est aussi une forme normale disjonctive (parmi d'autres).

Ouvrages de références

Champavère, J. (2007). Logique des propositions et logique des prédicats. *Note de cours*.

de La Guillonnière, G. S. (2012). Logique.

Gilles, F. (s. d.). *Logique propositionnelle.* Consulté sur http://cui.unige.ch/isi/icle-wiki/_media/cours:ffsi:4-logique-propositionnelle.pdf

Laurent, A. (2009). *Infz21, logiques du raisonnement valide.* Consulté sur https://laurent-audibert.developpez.com/Cours-Logique/?page=annexe-correction-travaux-diriges

Lucas, T., Berlanger, I., & Degauquier, V. (2014). *Initiation à la logique formelle. avec exercices et corrigés.*

Marie-Pierre, G. (2002). *Systèmes de preuves en logique des propositions.* Consulté sur http://spiralconnect.univ-lyon1.fr/webapp/course/course.html?id=129847&viewMode=visu&idChapter=136499

Rozière, P. (2004). Logique mathématique : introduction.

Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique

UNIVERSITE DE GUELMA
Faculté de Mathématiques et de
l'Informatique et des Sciences de la Matière
Département de l'Informatique

وزارة التعليم العالي والبحث العلمي

جـــامعة قـــالمـــة
كليـة الرياضيات و الإعلام الآلي و علو م المادة
قسم الإعلام الآلي

Polycopié des exercices

Master : Logique Mathématique
Et Fondements de l'Informatique

2016

Responsables du module Dr. KOUAHLA Zineddine

UNIVERSITE DE GUELMA

Avertissement

Ce document en est à sa première présentation. Malgré des soins particuliers consacrés à sa conception, l'auteur ne doute pas de la présence d'erreurs. Il prie le lecteur d'être indulgent, et le remercie de lui signaler les erreurs potentielles et des améliorations éventuelles. L'adresse électronique de l'auteur est kouahla.zineddine@univ-guelma.dz

Remerciements

L'auteur n'ayant pas inventé la logique, il s'est contenté d'exposer ce qu'il pense devoir être les notions fondamentales de la logique, en s'inspirant des travaux et œuvres de ses aînés. Sans qu'il en soit toujours fait mention explicitement, l'auteur a été guidé par les ouvrages cités en bibliographie, en particulier ceux de **Lalement**, **Gochet** et **Gribomont**, et **Quine**. L'auteur s'est aussi inspiré d'un cours de **Denis Lugiez**, chercheur au CRIN (Centre de Recherche en Informatique de Nancy).
Enfin, l'auteur remercie **Séridi Hamid** pour de nombreuses et fructueuses discussions autour de ce cours.

Logique est la science de la raison

SYLLABUS

Unit2 d'enseignement : UE5 (Systèmes intelligents2)
Matière : logique et fondements de l'informatique
Domaine/ Filière : 1er année Master Informatique académique.
Semestre : 1 et 2
Crédit : 06
Volume Horaire Hebdomadaire Total : 4h 30 mn
Cours Magistral : 3h
Travaux dirigés : 1,5h
Enseignant responsable : Dr KOUAHLA Zin-eddine
E-mail : zineddine.kouahla@univ-guelma.dz

Objectif du module

Pour ce module nous avons fixé deux objectifs :

1. Introduire les étudiants à la logique mathématique et en particulier a la méthode de la démonstration
2. Fournir aux étudiants les bases nécessaires afin de pouvoir comprendre le fonctionnement de la plupart des outils de démonstration automatique développés en particulier dans le monde académique, et éventuellement de coder eux-mêmes un tel outil

Naturellement, ce cours est nécessaire pour les étudiants qui poursuivent ensuite des travaux de recherche dans le domaine de la vérification, de la démonstration automatique ou de la réécriture, mais aussi dans les domaines connexes comme la sécurité de systèmes informatiques, les systèmes embarqués, les preuves assistées et plus généralement l'utilisation de systèmes formels.

Afin d'atteindre ces objectifs nous proposons d'utiliser au moins une des logiques suivantes :

- La logique du premier ordre (très utilisée pour formalisation et preuve dans des domaines comme l'IA et les bases de données)
- La logique équationnelle (exp : spécification et la validation de programmes séquentiels)
- La logique intuitionniste (qui donne des preuves constructives et les techniques de base de la démonstration automatique).

Semestre 1

Chapitre1 : Rappel mathématique

- Notion de terme, formule, connecteurs… etc

Chapitre2 : Logique de premier ordre

- Syntaxe (Construction de terme et de formules, variables libres et liées …)
- Sémantique (Modèle d'une formule, notion de structures, satisfiabilité, validité, substitution, équivalence, …)
- Complétude de la logique du premier ordre

- Théorie logique
- Formes normale (prenex, skolem, clausale)
- Résolution et programmation logique

Chapitre3 : Modèles de calcul

- Machines de Turing (non déterministes, à plusieurs rubans, alternantes, ..)
- Les automates finis
- Les RAM (Random Access Machine)

Semestre 2

Chapitre 1 : Calculabilité

- Fonctions récursives
- Calculabilité récursive
- Systèmes récursifs
- Décidabilité

Chapitre 2 : Décidabilité et incomplétudes

- Arithmétique et fonctions représentables
- Codage et preuves
- Problèmes indécidables

Chapitre 3 : Lambda Calcul

- Notion de terme du lambda calcul
- Fonctions normales
- Fonction récursives
- Lambda calcul typé

Chapitre 4 : Systèmes de types

- Système de type simple
- Déduction naturelle et systèmes de types

Chapitre 1 Introduction
(Logique propositionnelle & Logique des prédicats du premier ordre)
--Bilan--

1. Objectifs du chapitre

Ce chapitre est consacré à la partie logique propositionnelle et prédicats du premier ordre, qui correspond à la pratique de la déduction sur des énoncés ne pouvant prendre que les valeurs « vrai » ou « faux ».

2. Ce qu'il faut retenir

- Pourquoi a-t-on besoin d'un langage artificiel pour étudier le raisonnement? (Langage formel)
- Garantir que deux raisonnements de même forme soit simultanément correctes ou simultanément incorrectes

ON VOUDRAIT QUE L'EVALUATION DES RAISONNEMENTS SE FASSE D'UNE MANIERE MECANIQUE

- Une logique formelle consiste en un ensemble de formules, généralement des mots, avec une notion syntaxique de preuves, généralement des suites de formules obéissant à des règles de déduction, et une notion sémantique de valeur, déterminée à l'aide de réalisations convenables.
- Les formules de la logique propositionnelle $L_\bullet$ sont des assemblages de variables X_i à l'aide de connecteurs $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$.
- L'évaluation d'une formule de $L_\bullet$ se fait inductivement à partir de l'affectation de valeurs 0/1 aux variables.
- Une formule est dite valide (*resp*. satisfaisable) si sa valeur est 1 pour toute affectation (*resp*. pour au moins une affectation).
- On dit que H se déduit par coupure à partir de F et G si G est la formule $F \Rightarrow H$. Une preuve par coupure est une suite de formules dont chacune est soit un axiome pris dans une liste fixée de 14 types de formules, soit obtenue par coupure à partir de formules antérieures de la liste.
- Le théorème de complétude affirme qu'une formule de $L_\bullet$ est valide si et seulement si elle est prouvable.

- Le théorème de compacité affirme qu'un ensemble T de formules de L. est satisfaisable (c'est-`a-dire qu'il existe une affectation de valeurs 0/1 qui rende vraie chaque formule de T) si et seulement si tout sous-ensemble fini de T l'est.
- De nombreux problèmes peuvent être codés en un problème de satisfaisabilité (ou de validité) pour un ensemble de formules propositionnelles. Le théorème de Cook et Levin affirme le caractère NP-complet de l'ensemble SAT des formules satisfaisables.

Définitions de base

- *Alphabet* de la logique des prédicats est constitué de
- Ensemble dénombrable de symboles de prédicats à 0, 1, ou plusieurs arguments, notés p, q, r, ..., homme, mortel, père, ...
- Ensemble dénombrable de variables d'objets (ou variables d'individu), notées x, y, z, x_1, x_2, ...
- Ensemble dénombrable de fonctions à 0, 1, ou plusieurs arguments, notées f, g, ... , père-de, ...
- Quantificateurs $\forall$, $\exists$ (notations alternatives)
- Connecteurs False, ~, ∧ , ∨ , -> ainsi que les parenthèses de la logique propositionnelle

Notation.

- Q dénote un quantificateur quelconque, c-à-d ∀ ou ∃.

- Les fonctions à 0 arguments sont appellées constantes et sont notés sans parenthèses (a, b, ..., Socrate, ...). Même chose pour les prédicats à 0 arguments, qui ne sont rien d'autre que des variables propositionnelles.

Définition. *Terme*

L'ensemble des *termes* est le plus petit ensemble de mots construits sur l'alphabet de la logique des prédicats tel que

- Toute variable est un terme
- $f(t_1,...,t_n)$ est un terme si f est une fonction à n arguments et $t_1,...,t_n$ sont des termes

Définition. *Formule*.

- Si p est un prédicat à n arguments et $t_1,...,t_n$ sont des termes alors $p(t_1,...,t_n)$ est une *formule atomique*.

- L'ensemble FOR des *formules* (ou formules bien formées) de la logique des prédicats est alors défini de la même manière qu'en logique propositionnelle, en rajoutant une clause pour les quantificateurs :
- *(Q x A)* est une formule si *Q* est un quantificateur, x une variable et *A* une formule
- Une *expression* est un terme ou une formule.

Définition (Validité, satisfiabilité, modèle)

Soit *A*, une formule du calcul des prédicats sur une signature *(F, R)*.

- La formule est dite ***valide*** si sa valeur de vérité est vraie pour toute interprétation de la signature et tout environnement.
- La formule est dite ***satisfiable*** si sa valeur de vérité est vraie pour au moins une interprétation de la signature et un environnement. Une interprétation et un environnement qui rendent vraie la formule forment un ***modèle*** de la formule.
- La formule est dite ***insatisfiable*** si sa valeur de vérité est fausse pour toute interprétation de la signature et tout environnement.
- Ces définitions se généralisent à un ensemble de formules comme dans le cas propositionnel (interprétation comme une conjonction).

Résolution

La résolution traite un ensemble de clauses et déduit à partir de cet ensemble de nouvelles clauses en utilisant la règle de résolution :

$$\frac{C \vee D \qquad C' \vee \neg D}{C \vee C'}$$

Dans le cas du premier ordre, les clauses peuvent contenir des variables et donc chaque symbole de prédicat peut apparaître plusieurs fois, en concernant des termes différents.

Par ailleurs ces termes peuvent contenir des variables et dans ce cas la clause représente l'ensemble de toutes les instances possibles de la formule.

On part d'un ensemble de clauses (disjonction de littéraux vue comme un ensemble de littéraux). Chaque clause C dont les variables libres sont $x_1,\ldots,x_n$ est interprétée comme la formule universellement quantifiée $\forall\, x_1 \ldots x_n,\, C$ que l'on notera aussi $\forall(C)$.

L'unification sur les termes s'étend de manière naturelle à l'unification sur les littéraux. Deux littéraux L_1 et L_2 s'unifient s'ils sont de la forme $P(t_1,\ldots,t_n)$ et $P(u_1,\ldots,u_n)$ ou bien $\neg P(t_1,\ldots,t_n)$ et $\neg P(u_1,\ldots,u_n)$ et que le problème $(t_i \stackrel{?}{=} u_i)$ admet une solution. Dans ce cas on note unif(L_1,L_2)l'unificateur principal.

Règles pour la résolution

Les transformations de clauses en calcul des prédicats se font en utilisant trois règles :

1. *Factorisation* : si σ est l'unificateur principal de L_1 et L_2 alors :

$$\frac{L_1 \vee L_2 \vee C}{L_1[\sigma] \vee C[\sigma]}$$

2. *Renommage* : si σ est une bijection sur les variables alors :

$$\frac{C}{C[\sigma]}$$

3. *Résolution binaire* : si les clauses $L_1 \vee C$ et $\neg L_2 \vee C'$ n'ont pas de variables communes et si σ est l'unificateur principal de L_1 et L_2 alors :

$$\frac{L_1 \vee C \qquad \neg L_2 \vee C'}{C[\sigma] \vee C'[\sigma]}$$

Preuve par résolution

Les définitions de preuves par résolution sont une simple généralisation du cas propositionnel. Soit E un ensemble de clauses:

- une *déduction par résolution* à partir de E est une suite de clauses $C_1,\ldots,C_n$ telle que pour toute clause C_i dans cette suite
  - soit $C_i \in E$;
  - soit il existe une clause C_j telle que $j < i$ et telle que C_i soit le résultat de la règle de factorisation ou de renommage appliquée à C_j;
  - soit il existe deux clauses C_j et C_k telles que $j,k < i$ et telle que C_i soit le résultat de la règle de résolution appliquée à C_j et C_k.
- une *réfutation* de E est une *déduction par résolution* à partir de E qui contient la clause vide $\perp$;
- une *preuve par résolution* (on dit aussi une preuve par réfutation) d'une formule quelconque A à partir d'un ensemble de clauses E est une preuve par réfutation de l'ensemble de clauses $E \cup C_{\neg A}$ avec $C_{\neg A}$ une mise en forme clausale de la formule $\neg A$.

La règle de renommage sert juste à préparer les clauses pour appliquer la règle de résolution qui nécessite que les deux clauses ne partagent pas de variables. La règle de factorisation permet de faire disparaitre plusieurs littéraux lors d'une étape de résolution binaire.

Correction de la résolution

La correction de la méthode de résolution établit que si l'on peut déduire une clause C à partir d'un ensemble de clauses E alors la formule $\forall(C)$ est conséquence logique de l'ensemble des formules $\{\forall (D) | D \in E\}$

Une propriété clé est que si σ est une substitution alors $\forall (C) \models \forall (C[\sigma])$ qui se déduit du lien entre substitution et vérité :

$\mathrm{val}(e,C[\sigma])=\mathrm{val}(\{x\mapsto \mathrm{val}(e,\sigma(x))\},C)$

. Cette règle permet de ramener la correction de la résolution dans le cas du premier ordre à celle du cas propositionnel.

3. Exercice avec solution (Examen proposé en 2013)

L : est un langage de prédicats définit comme suit :

- *Alphabet x, y*
- *Constante 1*
- *Symbole b_0 et b_1*

On introduit les formules suivantes

$S_0 : S(l, b_0(l))$
$S_1 : \forall x, S(b_0(x), b_1(x))$
$S_2 : \forall x, S(x, y) \Rightarrow S(b_1(x), b_0(y))$

1. Soit une interprétation *I* dont le domaine *D* est formée des entiers naturels non nuls et dans laquelle les symboles sont interprétés de la manière suivante
$I_I = 1$
$(b_0)I\ (x) = 2 \times x$
$(b_1)I\ (x) = 2 \times x + 1.$
Donner deux interprétations différentes de la relation S qui satisfont les trois formules S_0, S_1 et S_2.
2. Pour chacun des couples de formules suivantes dire si elles sont unifiables et si oui donner l'unificateur principal :
   - $1 : S(b_1(x), y) ?= S(z, b_0(x))$
   - $2 : S(b_1(x), y) ?= S(x, b_0(z))$
   - $3 : S(b_1(x), l) ?= \neg S(b_1(l), z)$
   - $4 : S(b_1(x), l) ?= S(b_1(b_0(y)), x)$
3. En utilisant le système G, montrer le séquent
$S_0, S_1, S_2 \vdash \forall y, \exists x, S(x, b_1(b_0(y)))$.
4. Mettre les formules S_0, S_1 et S_2 en forme clausale.
5. En utilisant la méthode de résolution, montrer que la formule $\exists x, S(x, b_0(b_1(l)))$ est une conséquence logique des formules S_0, S_1 et S_2.
6. Toto se demande s'il peut prouver la formule ∃x, S(x, l) par résolution `a partir de S0, S1 et S2. Il essaie d'utiliser la même méthode, que constate-t-il ?
7. La formule $(S_0 \wedge S_1 \wedge S_2) \Rightarrow \exists x, S(x, l)$ est-elle valide ? Est-elle satisfiable ?

Correction

1. On peut prendre $SI = \{(x, y) \in N * \times N * | y = x + 1\}$ en effet, dans cette interprétation les formules deviennent
   $S_0 : 2 \times 1 = 1 + 1,$
   $S_1 : \forall x, 2x + 1 = 2x + 1,$
   $S_2 : \forall x\, y, y = x + 1 \Rightarrow (2 \times y) = 2x + 1 + 1$
   qui sont toutes vraies. Un autre modèle évident est si $S = N^* \times N^*$, c'est-à-dire que $S(x, y)$ est vrai pour tous les entiers non nuls.

   2.
   (a) unifiable $\{y \leftarrow b_0(x); z \leftarrow b_1(x)\}$
   (b) pas unifiable car il faudrait $x ?= b_1(x)$ qui n'a pas de solution
   (c) pas unifiable (formule et sa négation)
   (d) pas unifiable car il faudrait $x ?= l$ et $x ?= b_0(y)$

   3. (∀D) (∃D(x = b0(b0(y)))) (∀G(S1, x = b1(y))) hyp S0, S1, S(b0(b0(y)), b1(b0(y))), S2 ` S(b0(b0(y)), b1(b0(y))) S0, S1, S2 ` S(b0(b0(y)), b1(b0(y))) S0, S1, S2 ` ∃x, S(x, b1(b0(y))) S0, S1, S2 ` ∀y, ∃x, S(x, b1(b0(y))) 4. C0 : S(l, b0(l)) C1 : S(b0(x), b1(x)) C2 : ¬S(x, y) ∨ S(b1(x), b0(y)) 5. soit la formule F def = ∃x, S(x, b0(b1(l))) – On met ¬F en forme clausale on a ¬F ≡ ∀x, ¬S(x, b0(b1(l))) et donc C3 : ¬S(z, b0(b1(l))) – On unifie C2 et C3, avec {y ← b1(l); z ← b1(x); x ← x 0} on obtient C4 : ¬S(x 0 , b1(l)) on unifie C4 avec C1 avec {x ← l; x 0 ← b0(l)} et on obtient la clause vide. 6. Lorsque l'on met la formule ¬∃x, S(x, l) en forme clausale, on obtient ¬S(x, l), cette clause ne s'unifie `a aucune des clauses positives de C0, C1, C2, il n'y a donc pas moyen de d´eriver la clause vide en utilisant C3. Comme on a vu que les formules S0, S1, S2 avaient un mod`ele, il n'y a pas non plus de mani`ere de ´eriver la clause vide `a partir de C0, C1 et C2. 7. La formule (S0 ∧S1 ∧S2)⇒ ∃x, S(x, l) n'est pas valide (sinon on aurait une contradiction dans l'´etape pr´ec´edente ou encore il y aurait un mod`ele de (S0 ∧ S1 ∧ S2) (celui du successeur) qui ne valide pas ∃x, S(x, l), par contre cette formule est satisfiable si on prend le mod`ele dans lequel S est vrai partout.

4. Série des exercices proposés sans solution

Exercice 1.

P = {b, p}
b = « faire beau le matin »
p = « aller à la plage »

1. Ecrivez les formules correspondant aux énoncés suivants :
    a. S'il fait beau demain matin j'irai à la plage demain matin.
    b. Je suis allé à la plage, donc il faisait beau ce matin
    c. Je ne suis pas allé à la plage, donc il ne faisait pas beau ce matin
    d. Il a plu, donc je ne suis pas allé à la plage
2. Quel(s) énoncé(s) *(b, c, d)* peut-on déduire de l'énoncé a.

Exercices 2.

Extrait simplifié du « livre qui rend fou » de R. Smullyan.
Dans une île, on trouve deux sortes de personnes les menteurs et les sincères.
Les menteurs mentent toujours ; et les sincères disent toujours la vérité.

P = {b, a, c, d, e}
b = « Bob est sincère »,
d = « Denis est sincère »
a = « Alice est sincère »,
c = « Chloé est sincère »
e = « Elise est sincère »

A.
- **a.** Ecrivez les formules correspondant aux trois énoncés suivants
    - **i.** Alice dit « Bob dit toujours la vérité ».
    - **ii.** Bob dit « Alice est sincère ».
    - **iii.** Alice dit « Bob et moi nous sommes tous les deux sincères ou tous les deux menteurs ».
- **b.** Pouvez-vous déduire qu'Alice est sincère ?

B.
- **a.** Ecrivez les formules correspondant à l'énoncé suivant
    - **i.** Chloé dit « Je suis une menteuse ou Denis est sincère».
    - **ii.** Pouvez-vous déduite que Chloé est-elle une menteuse ? Que dire de Denis ?

C.
- **a.** Ecrivez les formules correspondant à l'énoncé suivant
    - **i.** Élise dit « Je suis une menteuse ».

ii. Qu'en pensez-vous ?

D. P = {f, g, h, i} f = « Franck est sincère », h = « Hugo est sincère » g = « Gabrielle est sincère » i = « Inès est sincère »

a. Ecrivez les formules correspondant aux énoncés suivants

i. Franck dit « Gabrielle et Hugo sont tous les deux sincères ou menteurs ».

ii. Gabrielle dit « Franck et Inès ne sont pas du même types : ils ne sont pas tous les deux menteurs ou tous les deux sincères ».

iii. Hugo dit « je suis un menteur ou Inès est une menteuse »

b. 4. Qu'en pensez-vous ?

Exercice 3.

Les définitions suivantes portent sur les entiers naturels (positifs ou nuls) :

- l'ensemble des fonctions : *G = {zéro, succ, plus} avec rang(zéro) = 0, rang(succ) = 1 et rang(plus) = 2.*
- l'ensemble des prédicats : *P = {pair, premier, inf_strict} avec rang(pair) = rang(premier) = 1 et rang(inf_strict) = 2.*
- l'ensemble des variables *: X = { x, y, z }.*

1. Les expressions ci-dessous sont ou non des formules syntaxiquement correctes, des termes syntaxiquement corrects, ou non ? Justifiez en détail
2. Pour chaque formule syntaxiquement correcte, déterminez si elle est close et la liste de ses variables libres.

1	*plus (x, plus(zéro, succ(zéro)))*
2	*succ(zéro)*
3	*((premier (x) ∨ inf_strict(zéro,y)) ∧ ¬pair(plus(y,z)))*
4	*∀x ∀z (premier(zéro) ⇒ (inf_strict(x, z) ∨ pair(plus(x, z))))*
5	*plus (plus (plus(plus(zéro, x), y), plus(x, plus(y, z)))*
6	*pair(x, 0)*
7	*x*
8	*(∀x premier (y) ⇒ ∃y premier (x))*
9	*∃x ∀y ∃x ∀z inf_strict (zéro, plus(x, plus(y, z)))*

Exercice 4.

Soit $G = \{vide, Ajouter, Retirer\} \cup \{a, \ldots, z\}$
avec $rang(vide) = 0$; $rang(Ajouter) = 2$; $rang(Retirer) = 2$; $rang(a) = 0 = \ldots = rang(z)$.
Soit $P = \{Estvide, EstLettre, Contient\}$
avec $rang(EstVide) = 1 = rang(EstLettre)$, $rang(Contient) = 2$

Soit $X = \{x1, x2, y1, y2, z1\}$.

1. Les expressions suivantes sont-elles des termes syntaxiquement corrects ou non ?
2. Les expressions suivantes sont-elles des formules syntaxiquement correctes ou non ? Justifiez en détail.
3. Pour chaque formule syntaxiquement correcte, déterminez si elle est close et la liste de ses variables libres.

1. $Ajouter(x_1)$
2. $EstLettre(Ajouter(x_1,y_1))$
3. $\forall x1\ Ajouter(x_2,d)$
4. $\forall x1\ EstVide(Ajouter(x_2,d))$
5. $\forall x2\ EstVide(Ajouter(x_2,d))$
6. $Contient(z_1)$
7. $EstLettre(Ajouter(Retirer(x_1,a),y_1))$
8. $\exists x1\ Contient(Retirer(x_1,d),d)$
9. $\exists x1\ Ajouter(x_1,d)$
10. $Ajouter(Ajouter(Ajouter(a,b),c),d)$
11. $Ajouter(Retirer(Ajouter(a,b),c),d) \wedge Retirer(Ajouter(v,e),l)$

Exercice 5.

Soit $G2= \{g_0, g_1, g_2\}$ *avec* $rang(g_0) = 0$; $rang(g1) = 1$; $rang(g_2) = 2$.
Soit $P2 = \{p_1, p_2\}$ *avec* $rang(p_1) = 2 = rang(p_2)$.
Soit $X = \{x_1, x_2, y_1, y_2, z_1\}$.

2. Définissez sur $L_2= P_2 \cup G_2$ une interprétation I_2 dont la sémantique correspond aux entiers relatifs (positifs, nuls ou négatifs) qui permet d'ordonner et de comparer les entiers. Remarque : l'interprétation des prédicats dans I_1 et I_2 doivent être différents.
3. Ecrivez un terme dont la valeur dans I_2 est 1 et un autre dont la valeur dans I_2 est -1.
4. Ecrivez un terme dont la valeur dans I_2 est 2 et un autre dont la valeur dans I_2 est -2.

4. Résumé du cours

Définition (axiomatique à la Hilbert).

Les schémas d'axiome de la logique des prédicats sont ceux de la logique propositionnelle, plus

$$\forall x\ A \to ((A)\{x\backslash t\})$$

$$(A)\{x\backslash t\} \to \exists x\ A$$

(où $\{x\backslash t\}$ est une substitution quelconque), et les règles d'inférence sont le Modus Ponens :

$$\frac{A \quad A \to B}{B}$$

Plus les deux règles pour les quantificateurs

$$\frac{A \to B}{A \to (\forall x\ B)}$$

S'il n'y a pas d'occurrence libre de x dans A et

$$\frac{A \to B}{(\exists x\ A) \to B}$$

S'il n'y a pas d'occurrence libre de x dans B

Note.

Il est possible de définir également la déduction. Nous l'omettons ici, car elle est plus complexe que celle de la logique propositionnelle (les deux règles d'inférence pour les quantificateurs peuvent seulement être appliquées à des axiomes logiques, et non à des axiomes non-logiques et des formules déduites à partir de ceux-ci). De plus, cette notion peut être réduite à la validité, par un théorème de la déduction similaire à celui de la logique propositionnelle.

Équivalences relatives aux quantificateurs

Ce sont les principes de base pour pouvoir mettre en forme normale.

Fermeture universelle :
Soit A sans occurrence libre de x. Alors

$$\vdash A \text{ ssi } \vdash \forall x\, A$$

Fermeture existentielle :
Soit A sans occurrence libre de x. Alors

$$A \text{ est satisfiable ssi } \exists x\, A \text{ est satisfiable.}$$

Quantification répetée :

$$\vdash (Q_1\, x\, Q_2\, x\, A) \leftrightarrow Q_2\, x\, A$$

Quantification sans variable libre :

soit A sans occurrence libre de x. Alors

$$\vdash (Q\, x\, A) \leftrightarrow A$$
$$\vdash (A)\{x\backslash t\} \leftrightarrow A$$

Renommage :

$$\vdash (Q\, x\, A) \leftrightarrow Q\, y\, (A)\{x\backslash y\}$$

Les deux cas où la distribution est possible :

$$\vdash (\forall x\, A) \wedge (\forall x\, B) \leftrightarrow \forall x\, (A \wedge B)$$
$$\vdash (\exists x\, A) \vee (\exists x\, B) \leftrightarrow \exists x\, (A \vee B)$$

Elargissement de la portée des quantificateurs :
Soit B sans occurrence libre de x. Alors

$$\vdash (Q\, x\, A) \wedge B \leftrightarrow Q\, x\, (A \wedge B)$$
$$\vdash (Q\, x\, A) \vee B \leftrightarrow Q\, x\, (A \vee B)$$

Note. Ces équivalences ne sont pas prouvables si x a des occurrences libres dans B :

p.ex. $(\exists x\, p(x)) \wedge \sim p(x)$ *n'est pas équivalent à* $\exists x\, (p(x) \wedge \sim p(x))$

Définition (forme normale prénexe).

Une formule A est en forme normale prénexe si elle est de la forme $Qx_1 \ldots Q_{xn}\, B$, *où* B est une formule sans quantificateurs.

La suite des quantificateurs est appelée préfixe, et B est appelée matrice.

Exemple 1 : mise en forme normale prénexe

Soit la formule $\exists x\, (\, p(x) \rightarrow \forall x\, p(x)\,)$:

Éliminer *->* :

$$\exists x\, (\, \sim p(x) \vee \forall x\, p(x)\,)$$

Renommer $\forall x\, p(x)$ *en* $\forall y\, p(y)$:

$$\exists x\, (\, \sim p(x) \vee \forall y\, p(y)\,)$$

Sortir $\forall y$:

$$\exists x\, \forall y\, (\, \sim p(x) \vee p(y)\,)$$

Note. Le renommage est essentiel ici ; p.ex. $\exists x\ (\sim p(x) \vee \forall x\ p(x))$ n'est pas équivalent à $\exists x\ \forall x\ (\sim p(x) \vee p(x))$

Exemple 2. mise en forme normale prénexe

Soit la formule $\sim(p(x) \rightarrow ((\exists y\ q(x,y)) \wedge \exists y\ r(y)))$:

Éliminer -> :

$\sim(\sim p(x) \vee ((\exists y\ q(x,y)) \wedge \exists y\ r(y)))$

Rentrer ~ :

$\sim\sim p(x) \wedge \sim((\exists y\ q(x,y)) \wedge \exists y\ r(y))$

$p(x) \wedge \sim((\exists y\ q(x,y)) \wedge \exists y\ r(y))$

$p(x) \wedge (\sim(\exists y\ q(x,y)) \vee \sim\exists y\ r(y))$

$p(x) \wedge ((\forall y \sim q(x,y)) \vee \sim\exists y\ r(y))$

$p(x) \wedge ((\forall y \sim q(x,y)) \vee \forall y \sim r(y))$

Renommer $\forall y \sim r(y)$ *en* $\forall z \sim r(z)$:

$p(x) \wedge ((\forall y \sim q(x,y)) \vee \forall z \sim r(z))$

Sortir $\forall y$ *et* $\forall z$:

$p(x) \wedge \forall y\ (\sim q(x,y) \vee \forall z \sim r(z))$

$\forall y\ (p(x) \wedge (\sim q(x,y) \vee \forall z \sim r(z)))$

$\forall y\ (p(x) \wedge \forall z\ (\sim q(x,y) \vee \sim r(z)))$

$\forall y\ \forall z\ (p(x) \wedge (\sim q(x,y) \vee \sim r(z)))$

Remarque.

Il est parfois possible d'être plus économique :

1. On peut économiser des quantificateurs en utilisant qu'on peut éliminer les quantificateurs qui « ne servent à rien » :
   La formule $\forall x\ \exists x\ (p(x) \vee q(x))$
   peut alors être simplifiée en : $\exists x\ (p(x) \vee q(x))$.
2. On peut économiser des variables (c-à-d éviter l'introduction de nouvelles variables) en utilisant la distribution de $\forall x$ sur $\wedge$ et de $\exists x$ sur $\vee$, (après avoir utilisé le renommage pour engendrer le même x partout) :
   La formule $\forall x\ (p(x) \wedge \forall y\ p(y))$
   est alors d'abord renommée en $\forall x\ (p(x) \wedge \forall x\ p(x))$
   et ensuite mis en forme normale prénexe : $\forall x\ (p(x) \wedge \forall x\ p(x))$.

Définition (Forme normale de Skolem)

Une formule est en *forme normale de Skolem* si elle est en forme normale prénexe et ne contient pas de quantificateur existentiel.

Algorithme de mise en forme normale de Skolem

***Entrée* :** une formule *A*

***Sortie* :** une formule en forme normale de Skolem

Début

mettre *A* en forme normale prénexe ;

pour tout quantificateur existentiel ∃x apparaissant dans A ***faire***

- Appliquer la substitution $\{x \backslash f(x_1,\ldots,x_n)\}$ à la matrice de A , où

$x_1,\ldots,x_n$ sont les quantificateurs *universels* précédant $\exists x$ dans

le préfixe de *A*

- *f* est une nouvelle fonction qui n'a pas encore été utilisée ;

supprimer ∃x du préfixe de *A*

fin pour tout

fin

Remarque. Si $n = 0$ on substitue par une constante.

Théorème. Pour toute formule d'entrée *A*, l'algorithme de mise en forme normale de Skolem s'arrête. Il retourne une formule *A'* en forme normale de Skolem telle que *A* est satisfiable ssi *A'* est satisfiable.
(la démonstration utilise que toutes les étapes préservent la satisfiabilité ; notons que la mise en forme normale prénexe préserve même l'équivalence logique)

Note. Dualement, on obtient l'équivalence de validité si on remplace les variables quantifiées universellement par une fonction des variables quantifiées existentiellement.

Exemples. mise en forme normale de Skolem

- Soit la formule $\exists x\ \forall y\ p(x,y)$:

remplacer x par la constante a : $\forall y\ p(a,y)$

- Soit la formule $\forall x\ \exists y\ p(x,y)$:

remplacer y par la fonction $f(x)$: $\forall x\ p(x,f(x))$

- Soit la formule $\exists u\ \forall x\ \exists y\ \forall z\ \exists t\ (\ p(x) \wedge q(y) \wedge r(x,z,t) \wedge s(y) \wedge k(u)\ )$:

remplacer u par la constante a : $\forall x\ \exists y\ \forall z\ \exists t\ (\ p(x) \wedge q(y) \wedge r(x,z,t) \wedge s(y) \wedge k(a)\ )$

remplacer y par la fonction $f(x)$: $\forall x\ \forall z\ \exists t\ (\ p(f(x)) \wedge q(f(x)) \wedge r(f(x),z,t) \wedge s(y) \wedge k(a)\ )$

remplacer t par la fonction $g(x,z)$: $\forall x\ \forall z\ (\ p(f(x)) \wedge q(f(x)) \wedge r(f(x),z,g(x,z)) \wedge s(y) \wedge k(a)\ )$

Définition (forme normale clausale).

Une formule est en *forme normale clausale* si elle est en forme normale de Skolem, fermée et sa matrice est en forme normale conjonctive propositionnelle.

Algorithme de mise en forme normale clausale

***Entrée* :** une formule *A*

***Sortie* :** une formule en forme normale clausale

Début

Pour tout variable *x* apparaissant libre dans *A* ***faire***

fermer *A* existentiellement : remplacer *A par* $\exists x\ A$

fin pour tout;

mettre *A* en forme normale de Skolem ;

mettre la matrice de *A* en forme normale conjonctive

fin

Notation.

- Comme toute variable est quantifiée universellement, nous pouvons éliminer le préfixe.

- Avec les mêmes définitions de littéral et clause qu'en logique propositionnelle nous pouvons appliquer la même convention notationnelle : une formule est représentée par un ensemble d'ensembles de littéraux.

Exemples

1. Soit la formule $\exists x\ (\ p(x) \rightarrow \forall x\ p(x)\ )$:

mise en forme normale prénexe : $\exists x\ \forall y\ (\ \sim p(x) \vee p(y)\ )$

mise en forme normale de Skolem : $\forall y\ (\ \sim p(a) \vee p(y)\ )$

... qui est aussi sa forme normale clausale ; l'ensemble de clauses associé est :

$\{\ \{\sim p(a)\}\ ,\ \{p(y)\}\ \}$

2. Soit la formule $\forall x\ \exists y\ (\ (p(x) \wedge q(x,y)) \vee r(z)\ )$:

fermer existentiellement : $\exists z\ \forall x\ \exists y\ (\ (p(x) \wedge q(x,y)) \vee r(z)\ )$

mettre en forme normale prénexe : la formule l'est déjà

mettre en forme normale de Skolem : $\forall x\ (\ (p(x) \wedge q(x,f(x))) \vee r(a)\ )$

mettre la matrice en forme normale conjonctive : $\forall x\ (p(x) \vee r(a)) \wedge (q(x,f(x)) \vee r(a))$

L'ensemble de clauses associé est $\{\ \{p(x)\ ,\ r(a)\},\ \{q(x,f(x))\ ,\ r(a)\}\ \}$

3. Série des exercices proposés

Exercice avec solution (Examen 2014)

On dira qu'une formule est en forme minimale si elle n'utilise que le connecteur $\Rightarrow$, et comme formule atomique $\perp$ et les variables propositionnelles.

1. Montrer que les formules $p \Rightarrow \perp$ *et* $\neg p$ sont logiquement équivalentes.
2. Sachant que les formules $p \Rightarrow q$ *et* $\neg p \vee q$ sont logiquement équivalentes, donner une formule équivalente à $p \vee q$ en forme minimale.
3. (a) Donner la table de vérité *de* $((p \Rightarrow (q \Rightarrow \perp)) \Rightarrow \perp$.
   (b) Comparer cette table et celle de $p \wedge q$.
   (c) En déduire une formule équivalente à $\neg p \wedge q$ qui soit en forme minimale.
4. Déduire des questions précédentes une fonction *min* qui transforme toute formule propositionnelle P (sans oublier $\top$) en une formule équivalente en forme minimale

Corrigé

1. Si p est vrai alors $p \Rightarrow \perp$ *et* $\neg p$ sont faux, si p est faux alors $p \Rightarrow \perp$ *et* $\neg p$ sont vrais, les formules sont équivalentes.
2. Une forme minimale pour $p \vee q$ *est* $(p \Rightarrow \perp) \Rightarrow q$ (qui est équivalent à $\neg p \Rightarrow q$)
3. (a)

p	q	$p \Rightarrow (q \Rightarrow \perp)$	$(p \Rightarrow (q \Rightarrow \perp)) \Rightarrow \perp$
F	-	V	F
V	F	V	F
V	V	F	V

   (b) c'est la même table que $p \wedge q$
   (c) $((p \Rightarrow \perp) \Rightarrow q \Rightarrow \perp) \Rightarrow \perp$, une autre forme possible est $(q \Rightarrow p) \Rightarrow \perp$
4.

$min(\perp) = \perp$
$min(>) = \perp \Rightarrow \perp$
$min(p) = p$ *si* p *var. prop.*
$min(\neg P) = min(P) \Rightarrow \perp$
$min(P \vee Q) = (min(P) \Rightarrow \perp) \Rightarrow min(Q)$
$min(P \wedge Q) = (min(P) \Rightarrow (min(Q) \Rightarrow \perp)) \Rightarrow \perp$
$min(P \Rightarrow Q) = min(P) \Rightarrow min(Q)$

Exercice 1 : Preuve par construction

De nombreux théorèmes stipulent qu'un objet particulier existe.
Une technique pour prouver un tel théorème est de montrer comment construire cet objet.
Cette technique est appelée *preuve par construction*.

Exemple
Un graphe non orienté G est défini par le couple (S, A), S étant un ensemble de sommets, $A \subset S^2$ un ensemble d'arrêtes. On appelle le degré d'un sommet le nombre d'arrêtes reliées à ce sommet. Un graphe est dit k-régulier si tous ses sommets sont de degré k.
Théorème : Pour tout entier n pair, $n > 2$, il existe un graphe 3-régulier composé de n sommets.
Preuve : Exercice.

Exercice 2. Preuve par contradiction

Cette technique consiste à supposer que le théorème est faux, puis à montrer que cette hypothèse conduit systématiquement à une conséquence fausse, appelée contradiction.

Exemple
Un nombre x est rationnel s'il existe deux entiers m et n tels que $x = m / n$. Par exemple, $3 / 4$ est un nombre rationnel.
Un nombre non rationnel est appelé un nombre irrationnel.

Théorème : $\sqrt{2}$ est irrationnel.
Preuve : Exercice.

Théorème : Soit R une relation binaire définie sur un ensemble fini A, et soient a et b deux éléments de A. S'il existe un chemin de a vers b dans R, alors il existe un chemin de longueur au plus $|A|$.
Preuve : exercice.

Exercice 3. Preuve par récurrence, ou induction

Cette technique de preuve permet de montrer que tous les éléments d'une suite infinie vérifient une certaine propriété P. Elle comporte deux étapes :
- étape de base : on montre que $P(k)$ est vraie ;
- pas d'induction : pour $i \geq k$, on montre que si $P(i)$ est vraie, alors $P(i+1)$ est également vraie.

$P(n)$ est alors vraie, pour tout $n \geq k$, parce que $P(k)$ est vraie (étape de base), donc $P(k+1)$ est vraie (pas d'induction pour $i = k$), donc $P(k+2)$ est vraie, etc.
Exemple
Théorème : $n^4 - 4n^2$ est divisible par 3.
Preuve : exercice.

Exercice 4. Montrer que, dans la liste des axiomes pour L_Σ, on peut remplacer les axiomes pour l'égalité par les formules $x = x$ et $x = y \Rightarrow F(x) \Leftrightarrow F(y)$ avec x et y sont libres pour z dans $F(z)$.

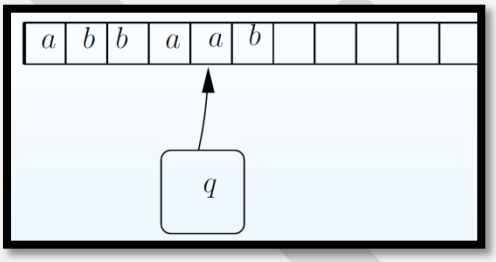
4. Machine de Turing

Alan Turing inventa cette machine abstraite en 1936 pour définir la notion de "fonction calculable".

- Toute tâche exécutée par une machine de Turing MT peut l'être sur un ordinateur ... et vice-versa !
- Peut simuler n'importe quel automate fini, automates à pile et même n'importe quel programme exécutable sur un ordinateur
- Il existe une MT universelle capable de simuler toutes les autres MT : programme comme donnée

Définition.

Une MT est composée d'une unité de contrôle, d'un ruban infini divisé en cases et d'une tête de lecture/ écriture.



En fonction de ce qu'elle lit sur la case courante et de son état courant (conformément à sa relation de transition δ), elle :

1. Ecrit un symbole sur sa case courante
2. Déplace la tête de lecture/ écriture d'une case (à droite, à gauche)
3. Change d'état.

Une Machine de Turing M est donnée par

$$(Q, \sum, \Gamma, \delta, q_0, F) \text{ où :}$$

• Q est un ensemble fini d'états.
• $\sum$ est l'alphabet (fini) d'entrée
• Γ est l'alphabet (fini) du ruban, tel que $\sum \subseteq \Gamma$
• $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{G,D, I\}$ est la fonction de transition.
• q_0 est l'état initial.
• $F \subseteq Q$ est l'ensemble des états accepteurs.
• On suppose qu'aucune transition n'est définie dans les états accepteurs.

• On considère un symbole B $\in \Gamma \setminus \sum$ particulier dit blanc.

5. Machine de Turing à plusieurs rubans

- Une MT à k rubans, avec $k \geq 1$, est composée d'une unité de contrôle, de k rubans infinis divisé en cases, de k têtes de lecture/ écriture.
- Chaque ruban a sa propre tête de lecture.
- En fonction de ce qu'elle lit sur les k cases courantes et de son état courant (conformément à sa relation de transition δ), elle :
  1. écrit un symbole sur la case courante de chaque ruban
  2. déplace chacune des têtes de lecture/ écriture d'une case (à droite, à gauche)
  3. change d'état

$$(Q, \sum, \Gamma, \delta, q_0, F) \text{ où :}$$

Théorème de Rice

Soit $\sum = \{0, 1\}$.

Une propriété non-triviale sur les machines de Turing est un sous-ensemble $P \subseteq \sum^*$ tel que :

1. Pout toutes machines M et N :
$$L(M) = L(N) \Rightarrow [M \in P \Leftrightarrow N \in P].$$
2. Il existe $M_0, M_1 : M_0 \in P$ et $M_1 \notin P$.

3. Haut niveau (Machines RAM)

Le modèle de la machine de Turing peut paraître extrêmement rudimentaire.
Il n'en demeure pas extrêmement puissant, et capable de capturer la notion de calculable en informatique.

Définition & Fonctionnement

Un modèle de calcul qui ressemble beaucoup plus aux langages machine actuels, et à la façon dont fonctionnent les processeurs actuels.
Une machine RAM possède des registres qui contiennent des entiers naturels (nuls si pas encore initialisés).
Les instructions autorisées dépendent du processeur que l'on veut modéliser, mais elles incluent en général la possibilité de :

1. Copier le contenu d'un registre dans un autre ;
2. L'adressage indirect : récupérer/écrire le contenu d'un registre dont le numéro est donné par la valeur d'un autre registre ;
3. Effectuer des opérations élémentaires sur un ou des registres, par exemple additionner 1, soustraire 1 ou tester l'égalité à 0 ;

4. Effectuer d'autres opérations sur un ou des registres, par exemple l'addition, la soustraction, la multiplication, division, les décalages binaires, les opérations binaires bit à bit.

4. Haut niveau (Machines RISC (Reduced Instruction Set))

Une machine RISC est une machine RAM dont les instructions sont uniquement de la forme :

1. $x_0 \leftarrow 0$;
2. $x_0 \leftarrow x_0 + 1$;
3. $x_0 \leftarrow x_0 \quad 1$;
4. ***if*** $x_0 = 0$ ***then aller à l'instruction numéro*** j ;
5. $x_0 \leftarrow x_i$;
6. $x_i \leftarrow x_0$;
7. $x_0 \leftarrow x_{xi}$;
8. $x_{x0} \leftarrow x_i$.

Définition Théorème. Toute machine RISC peut être simulée par une machine de Turing. (et réciproquement).

Exercice (Démonstration du Théorème)

5. Exercice proposés

Exercice avec solution

1. Construire une machine de Turing acceptant le langage $\{ucu^{-} \mid u \in \{a, b\}^*\}$.
2. Construire une machine de Turing acceptant le langage $\{u \in \{a, b\}^* \mid \ |u|_a = |u|_b\}$.
3. construire une machine de Turing calculant $n + 1$ pour n donné en binaire sur le ruban d'entrée.
4. En utilisant ce qui précède, décrire une méthode qui permettrait de calculer $n+m$ en binaire (*n et m donnés sur le ruban*).

Solution

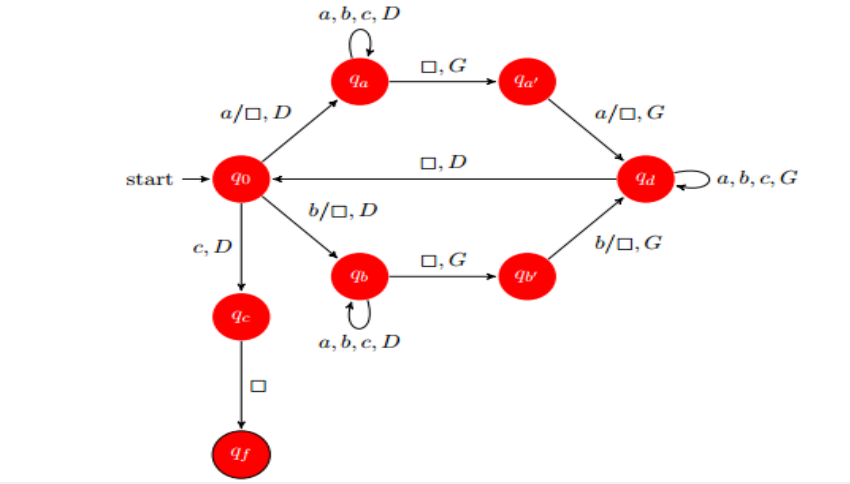


Figure 1. Un mot, un séparateur (c), un mot miroir

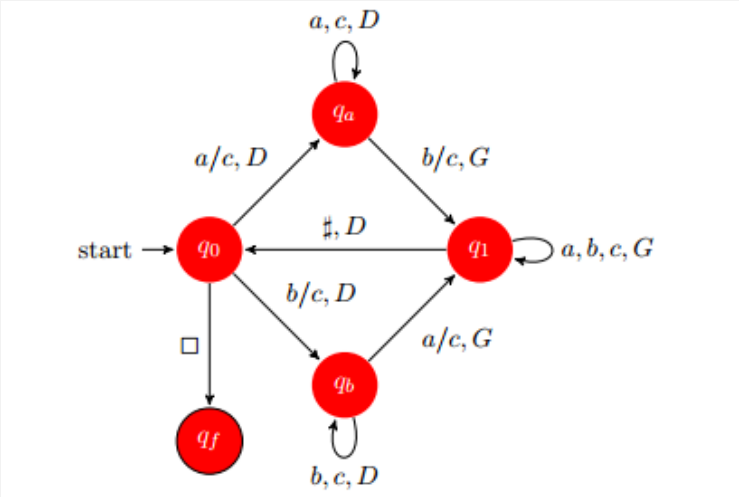


Figure 2. Autant de a que de b

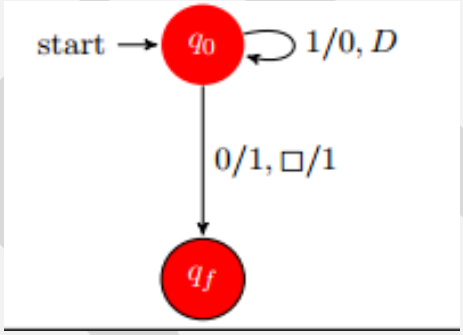
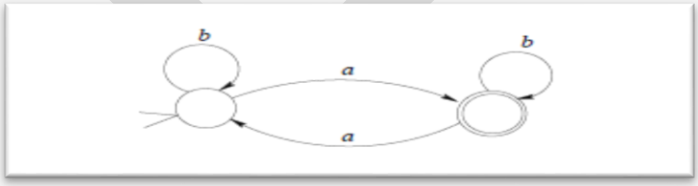


Figure 3. Successeur binaire

Exercice 1. On considère l'automate défini par le schéma suivant :

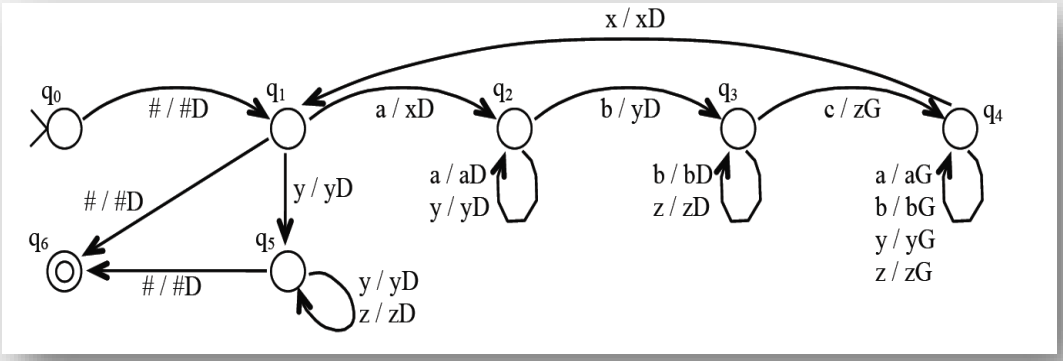


1. Construire une machine de Turing qui « mime » le fonctionnement de cet automate.
2. Généralisez votre réponse en indiquant un moyen de transformer tout automate déterministe fini en une machine de Turing.

3. Qu'en est-il des automates indéterministes ? Des automates à pile ?

Exercice 2.

1. Construisez une machine de Turing qui accepte $L = \{a^n b^n c^n \mid n \in N\}$.
2. Quel est le langage reconnu par la machine suivante ?



Exercice 3.

1. Construisez une machine de Turing qui calcule la fonction $add : \boldsymbol{N}^2 \rightarrow \boldsymbol{N}$
   telle que $add\ (m,n) = m + n$, m et n
Étant représentés en unaire.
2. Construisez une machine de Turing qui calcule la fonction $f : \boldsymbol{N} \rightarrow \boldsymbol{N}$
   telle que $f(x) = x + 1$,
x étant représenté en binaire.
3. Soit $\Sigma = \{a, b\}$. Construisez une machine de Turing calculant la fonction f : $\Sigma^* \rightarrow \Sigma^*$ qui décale tout mot w d'une case vers la droite.
4. Construisez une machine de Turing calculant la fonction $f : \Sigma^* \rightarrow \Sigma^*$ qui à tout mot w associe le mot miroir w^R.

Exercice 4.

1. Construisez la machine de Turing C sous forme de combinaison de machines de Turing élémentaires qui effectue une copie :

$$(q_0, \underline{\#}\, \sigma_1 \ldots \sigma_n) \vdash C^* (q_f, \underline{\#}\sigma_1 \ldots \sigma_n \# \sigma_1 \ldots \sigma_n)$$

2. Déroulez C sur l'exemple $(q_0, \underline{\#}aba)$

Exercice 5.

1. Soit $\Sigma = \{a, b\}$.
   a) Construire une machine de Turing qui prend un mot u sur son ruban d'entrée et écrit uu sur un ruban de sortie.
   b) Construire une machine de Turing qui décide le langage $L = \{w \in \Sigma^* \mid w = w^R\}$.
   c) Construire une machine de Turing qui décide le langage $L = \{w \in \Sigma^* \mid |w|_a = |w|_b\}$.
2. Soit $\Sigma = \{0, 1\}$.
   a) Construire une machine de Turing qui
      - ne s'arrête pas si le mot en entrée contient le symbole 1,
      - écrit en binaire, sur son ruban de sortie, le nombre de 0 du mot en entrée avant de s'arrêter.
   b) Construire une machine de Turing qui convertit en unaire un entier entré en binaire.

c) Construire une machine de Turing qui convertit en binaire un entier entré en unaire.

Chapitre 4 Calculabilité & Décidabilité

1. Motivation

Cette partie présente les principaux résultats de la calculabilité. En d'autres termes, l'étudiant doit chercher à comprendre la puissance des programmes informatiques.

On y prouve que certains problèmes peuvent se résoudre informatiquement et que certains problèmes ne peuvent pas l'être.
Tout d'abord il convient de comprendre que les problèmes auxquels nous allons nous intéresser ne sont pas vraiment des problèmes pour lesquels on ne connaît pas de solution, mais, et c'est en encore beaucoup plus fort, des problèmes pour lesquels on sait qu'il est impossible de produire une solution algorithmique.

Pourquoi s'intéresser à comprendre les problèmes qui ne peuvent pas être résolus ?
Premièrement, parce que comprendre qu'un problème ne peut pas être résolu est utile. Cela signifie que le problème doit être simplifié ou modifié pour pouvoir être résolu.
Deuxièmement, parce que tous ces résultats sont culturellement très intéressants et permettent de bien mettre en perspective la programmation, et les limites des dispositifs de calculs, ou de l'automatisation de certaines tâches, comme par exemple la vérification.

2. Résumé du cours

a. Langages et problèmes décidables

1. Problèmes de décision

Définition
Un problème de décision P est la donnée d'un ensemble E, que l'on appelle l'ensemble des instances, et d'un sous-ensemble E^+ *de* E, que l'on appelle l'ensemble des instances positives

2. Langages décidables

On rappelle la notion de langage décidé, qui invite à introduire la notion de langage décidable.

Définition (Langage décidable)
Un langage $L \subset M^*$ est dit décidable s'il est décidé par une certaine machine de Turing.
Un langage ou un problème décidable est aussi dit récursif.
Un langage qui n'est pas décidable est dit indécidable.

b. Problèmes semi-décidables

Le problème HALTING − PROBLEM est toutefois semi-décidable dans ce sens

Définition (Langage semi-décidable)
Un langage $L \subset M^*$ est dit semi-décidable s'il correspond à l'ensemble des mots acceptés par une machine de Turing M.

Corollaire. Le langage universel HALTING − PROBLEM est semi-décidable.

Démonstration
Pour savoir si on doit accepter une entrée qui correspond au codage <M> d'une machine de Turing M et au mot w, il suffit de simuler la machine de Turing M sur l'entrée w.
On arrête la simulation et on accepte si l'on détecte dans cette simulation que la machine de Turing M atteint un état accepteur. Sinon, on simule M pour toujours.

Un langage semi-décidable est aussi dit récursivement énumérable.

Corollaire. R $\not\subset$ RE.

Démonstration
L'inclusion est par définition. Puisque HALTING − PROBLEM est dans RE et n'est pas dans R, l'inclusion est stricte.

3. Exercices proposées

Exercice (Calculabilité)
Monter que la division euclidienne est calculable.

Exercice (Calculabilité)
Montrer que la fonction $x \rightarrow x^2$ est calculable.

Exercice (Calculabilité)
Montrer que si

$$f : N^n \rightarrow N$$
$$g : N^2 \rightarrow N$$

sont calculables alors

$$h : N^{n+1} \rightarrow N \text{ avec } h(x_1, \cdots, x_{n+1}) = g(f(x_1, \cdots, x_n), x_{n+1})$$

est calculable.

Exercice (Décidabilité des classes de langages)

1. Les langages rationnels sont-ils décidables ? Justifiez.
2. Les langages algébriques sont-ils décidables ? Justifiez.

Exercice (Problèmes indécidables à propos des grammaires algébriques)

1. Montrez que le problème suivant est indécidable :
   Soit G une grammaire algébrique. Est-ce que $L(G) = \Sigma^*$?
2. Montrez que le problème suivant est indécidable :
   Soient G_1 et G_2 deux grammaires algébriques. Est-ce que $L(G_1) \cap L(G_2) = \emptyset$?

Exercice (Problèmes indécidables à propos des machines de Turing)

1. Soient M_1 et M_2 deux machines de Turing. Montrez que déterminer si M_1 et M_2 acceptent le même langage est un problème indécidable.
2. Soit L un langage accepté par une machine de Turing, et soit L^R le langage constitué des mots de L vus dans un miroir.
   Par exemple, L = {a, ab, abb, abbb}, L^R = {a, ba, bba, bbba}.
   Montrez que le problème qui consiste à montrer que $L = L^R$ est un problème indécidable.

Aller plus loin

Exercice

Soit $E \subset \mathbb{N}$ un ensemble récursivement énumérable, énuméré par une fonction calculable f strictement croissante.
Montrer que E est décidable.

Exercice

En déduire que tout sous-ensemble récursivement énumérable infini de $\mathbb{N}$ contient un ensemble décidable infini.

Exercice

Soit $E \subset \mathbb{N}$ un ensemble décidable.
Montrer qu'il est énuméré par une fonction calculable f strictement croissante.

Exercice

Soit $A \subset \mathbb{N}^2$ un ensemble décidable de couples d'entiers.
On note $\exists A$ pour la (première) projection de A, à savoir le sous-ensemble de $\mathbb{N}$ défini par
$\exists A = \{x | \exists y \in \mathbb{N} \text{ tel que } (x, y) \in A\}$.

Chapitre 5 Récursivité

1. Résumé du cours

Les définitions récursives sont omniprésentes en informatique.
Elles sont présentes à la fois dans les langages de programmation, mais aussi présentes dans de nombreux concepts que l'on manipule

Les termes *récursif*, *récursion*, *récursivité* ont la même racine latine que *récurrence* : *currere* = *courir*, et par là : *recurrere* = *courir en arrière* qui a donné également *recourir*, *recours* = *action de faire appel* (en justice).

- Le terme *récurrence* apparaît avec Poincaré dans *La Science et L'hypothèse* au tout début du 20è siècle. On parle dès lors de formule de récurrence, et de raisonnement par récurrence pour signifier le raisonnement par induction de Pascal.
- La *récursivité* n'apparaît dans le Larousse qu'en 1968 avec les progrès de l'informatique et des algorithmes récursifs : la notion actuelle de fonction (ou de procédure) *récursive* rencontrée en programmation se rattache à celle définie par Skolem (1923) puis Gödel (1926) sans en avoir exactement le même sens.

Les fonctions récursives primitives, récursives élémentaires et récursives générales :

On suppose ici que l'ensemble ***N*** des entiers naturels est construit ou accepté dans son sens intuitif. Selon Kleene, les fonctions de $\boldsymbol{N}^p$ dans ***N***, $p \geq 1$, établies comme effectivement calculables depuis les premiers travaux de Skolem et Gödel sur le sujet, dites *récursives primitives* (appellation du à Skolem) sont les suivantes :

1. La fonction successeur (incrémentation) Incr qui à tout entier naturel x associe l'entier immédiatement supérieur x^+. C'est donc $x + 1$, mais tant que l'addition n'est pas définie...
2. Les fonctions constantes $F_c(x_1, ..., x_p) = c$ *(entier)* dont la fonction nulle.
3. Les fonctions projections $P_i(x_1, ..., x_p) = x_i$.
   Pour deux variables : $P_2(x_1,x_2) = x_2$, pour une variable : $P_1(x_1) = x_1$, c'est *l'application identique*.
4. Les fonctions obtenues par récurrence simple sur une des variables entières, par exemple x_1 : on suppose g récursive primitive, alors f définie par l'algorithme ci-dessous l'est aussi :
   a. initialisation : $f(0, x_2, ..., x_p) = k$, k donné dans ***N*** Raisonnement par induction
   b. hérédité (ou *induction*) : $f(n+1, x_2, ..., x_p) = g(f(n, x_2, ..., x_p), x_2, ..., x_p)$
5. Les fonctions obtenues par composition (on dit aussi par *superposition*) : si f et g sont récursives primitives, alors la fonction composée h définie par

   $h(x_1, ..., x_p) = f(x_1, ...,g(x_1, ..., x_p),..., x_p)$ l'est aussi.

6. Les fonctions obtenues par l'opération de minimum (dû à Kleene) :
   si $R(x_1, ..., x_p)$ entre éléments de ***N***, on désigne par $(\mu x)R(x_1, ..., x_p)$, *mu* comme minimum, le plus petit entier naturel x réalisant $R(x_1, ..., x_p)$, ce qui permet de construire de nouvelles fonctions.

La nature récursive de ces fonctions (au sens usuel de fonction se définissant en faisant appel à elle-même) n'apparaît peut-être pas clairement.
Étudions-les à travers des exemples plus concrets, *fonctions récursives élémentaires* :

La fonction prédécesseur *Dec* (décrémentation, antécédent)

Si x est un entier naturel, son antécédent est 0 s'il est nul, sinon, c'est $x - 1$. Or, $(x+1)-1=x$. On peut donc la définir très simplement au moyen de la fonction successeur par :

Ce qui peut s'écrire $Dec(0) = 0$ et $Dec(n+1)=n$
Il vient ainsi $Decr(1)=Decr(0+1) =0$, $Decr(2)=Decr(1+1)=1$, et ainsi de suite.
L'algorithme correspondant procède par itération (boucle).

La fonction addition *Add* :

Additionner y à x, c'est incrémenter x en décrémentant y jusqu'à obtenir 0.

On peut la définir par récurrence :

$Add(x,0) = P_1(x)$ (i.e. $= x$, condition d'arrêt)
$Add(x,y) = Incr(Add(x,Decr(y))$

- Par exemple : $Add(2,3) = Incr(Add(2,2)) = Add(2,2) + 1 = [Incr(Add(2,1)) + 1] + 1$
  $= [[Add(2,0) + 1] + 1] + 1$
  $= [[2 + 1] + 1] + 1 = [3 + 1] + 1 = 4 + 1 = 5.$

Et on pose plus simplement : $Add(x,y) = x + y$.

La fonction soustraction Sous :

La soustraction dans $\boldsymbol{N}$ définie par : $S(x,y) = x - y$ si $x \geq y$ et 0 sinon, peut être définie par :

$Sous(x, 0) = x$ et $Sous(x,y) = Decr(Sous(x,Decr(y))$

- Par exemple : $Sous(5,3) = Decr(Sous(5,2)) = Decr(Decr(Sous(5,1)))$
  $= Decr(Decr(Decr(Sous(5,0)))) = Decr(Decr(Decr(5)))$
  $= Decr(Decr(4)) = Decr(3) = 2$

La fonction multiplication *M* :

$M(x,y) = x \times y$ Elle sera définie par :

$M(x,0) = N(x)$ (fonction nulle : condition d'arrêt)
$M(x,y) = A(M(x,D(y)),P_1(x,y))$, c'est à dire $x(y - 1) + x$: *appel récursif*

Exemple (Arbres ordonnés)

Une alternative naturelle serait de présenter les arbres ordonnés par une définition récursive : un arbre ordonné est soit vide, soit réduit à un sommet (une racine), soit constitué d'un sommet (une racine) et une liste (ordonnée) d'arbres ordonnés (ses fils).

Dans ce chapitre, nous nous attardons sur les définitions inductives d'ensembles et de fonctions, qui permettent de donner un sens à des définitions récursives. Nous discutons, par ailleurs comment il est possible de faire des preuves sur des structures définies inductivement, en introduisant les preuves par induction structurelle.

Raisonnement par récurrence sur N

Théorème Soit $P(n)$ un prédicat (une propriété) dépendant de l'entier n.
Si les deux conditions suivantes sont vérifiées :
(B) $P(0)$ est vrai ;
(I) $P(n)$ implique $P(n + 1)$ pour tout n ;

Alors pour tout entier n, $P(n)$ est vrai.

Démonstration:
Le ***raisonnement*** se fait par l'absurde. Considérons $X = \{k \in N | P(k) \text{ est faux}\}$.
Si X est non vide, il admet un plus petit élément n.
D'après la condition *(B)*, $n \neq 0$, et donc $n-1$ est un entier, et $P(n - 1)$ est vrai par définition de X. On obtient une contradiction avec la propriété *(I)* appliquée pour l'entier $n-1$.

Pour faire une preuve par récurrence, on établit donc une propriété en 0 (cas de base), et on établit que la propriété est héréditaire, ou inductive : $P(n)$ implique $P(n+ 1)$ pour tout n.
Le concept de preuve inductive généralise cette idée à d'autres ensembles que les entiers, à savoir aux ensembles qui se définissent inductivement.
Formellement, tout cela se justifie par le théorème qui suit.

Définition. (Définition inductive) Soit E un ensemble. Une définition inductive d'une partie X de E consiste à se donner

- un sous ensemble non vide B de E (appelé ensemble de base)
- et d'un ensemble de règles R : chaque règle $r_i \in R$ est une fonction (qui peut être partielle) r_i de $E_{ni} \rightarrow E$ pour un certain entier $n_i \geq 1$

Théorème (Théorème du point fixe) *A* une définition inductive correspond un plus petit ensemble qui vérifie les propriétés suivantes
(B) il contient $B : B \subset X$;
(I) il est stable par les règles de R : pour chaque règle $r_i \in R$, pour tout
$x_1, \cdots, x_{ni} \in X$, on a $r_i(x_1, \cdots, x_{ni}) \in X$.

On dit que cet ensemble est alors défini inductivement.

Démonstration Soit F l'ensemble des parties de E vérifiant *(B) et (I)*.
L'ensemble F est non vide car il contient au moins un élément : en effet, l'ensemble E vérifie les conditions *(B) et (I) et donc* $E \in F$.
On peut alors considérer X défini comme l'intersection de tous les éléments de F.

Exercice proposés (Exemples d'applications)

Exercice 1.
Montrez que les fonctions suivantes sont primitives récursives :
a) la fonction *factorielle*(*n*)
b) la fonction *pgcd*(m, *n*)
c) le prédicat *prem*(*n*) qui vaut *1* si *n* est un nombre premier, *0* sinon

Exercice 2.
Montrer que les fonctions suivantes sont récursives primitives.
1. $x \rightarrow 0$
2. les fonctions constantes égales à *3* d'arité *0, 1 et 2 ;*
3. la fonction addition : $(x; y) \rightarrow x + y$;
4. la fonction multiplication : $(x; y) \rightarrow x * y$;
5. la fonction exposant : $(x; y) \rightarrow xy = x'' y$;
6. la fonction prédécesseur *Pred* : $x \rightarrow (x - 1)$ si $x > 0$
   0 sinon ;
7. la fonction différence tronquée *Diff* : $(x; y) \rightarrow (x-y)$ si $x \geq y$
   0 sinon ;

Exercice 3.
Montrer que le prédicat x > y est récursif primitif.

Exercice 4.
1. Montrer que les constructeurs suivants sont récursifs primitifs (c'est à dire que s'ils sont utilises sur des fonctions récursives primitives du bon type alors les fonctions qu'ils construisent sont bien récursives primitives).

2. Pour $k \in N$, si P est un prédicat d'arite k, f *et* g sont des fonctions d'arite k, alors Si$(P; f; g)$ est la fonction h d'arité k telle que lorsque P est vrai $h = f$ et lorsque P est faux $h = g$.
3. Pour $k \in N$, si f est une fonction d'arite $k + 1$ alors Somme(f) est la fonction h d'arité $k + 1$ telle que

$$h(y, x_1, \ldots, x_k) = \sum_{i=0}^{y} f(i, x_1, \ldots, x_k)$$

4. Pour $k \in N$, si f est une fonction d'arité $k + 1$ alors *Produit(f)* est la fonction h d'arité $k + 1$ telle que

$$h(y, x_1, \ldots, x_k) = \prod_{i=0}^{y} f(i, x_1, \ldots, x_k)$$

Exercice 5. (Fonctions µ-récursives)

1. Montrez que la fonction logarithme est une fonction µ-récursive.

2. Utilisez l'opérateur de minimisation pour définir les fonctions :

$$quot(m,n) = \begin{cases} \left\lfloor \frac{m}{n} \right\rfloor & \text{si } n > 0 \\ 0 & \text{sinon} \end{cases}$$

$$\text{mod}(m,n) = \begin{cases} m - \left\lfloor \frac{m}{n} \right\rfloor . n & \text{si } n > 0 \\ 0 & \text{sinon} \end{cases}$$

3. On suppose connu le prédicat *prem*(n) qui vaut i si n est un nombre premier, 0 sinon. Montrez que la fonction *nprem(n)* qui renvoie le $n^{ème}$ nombre premier est une fonction µ-récursive.

Chapitre 6 Lambda Calcul

1. Résumé du cours

Pour l'informaticien, l'étude du λ-calcul (prononcer lambda-calcul) permet de comprendre sur un langage minimal des concepts théoriques qui pourront par la suite se reporter sur des langages de programmation plus riches. Par exemple, nous pourrons raisonner sur la terminaison des programmes, sur l'équivalence entre deux programmes, sur les relations entre la programmation et la logique

Syntaxe

On donne habituellement la syntaxe du λ-calcul en disant que c'est l'ensemble des expressions e formées de la façon suivante :

$$\boxed{e ::= \lambda x.\ e | e_1 e_2 | x}$$

Pour une présentation un peu plus précise, on suppose l'existence d'un ensemble V infini de variables (que nous noterons $x, y, \ldots, x_i, y_i, f, g$) et l'ensemble des λ-termes est constitué de la façon suivante :

- Si x est une variable et e est un λ-terme déjà construit, alors $\lambda x.\ e$ est un λ-terme, nous appellerons un tel λ-terme une abstraction.
- Si e_1 et e_2 sont des λ-termes déjà construits alors $e_1\ e_2$ (la juxtaposition des termes) est un λ-terme, nous appellerons un tel λ-terme une application.
- Si x est une variable, alors c'est également un λ-terme.

Quelques exemples fréquemment rencontrés de λ-termes :

a) $\lambda x.\ x$ *(on l'appelle I),*
b) $\lambda x.\ \lambda y.\ x$ *(K),*
c) $\lambda x.\ x\ x$ *(Δ),*
d) $\lambda x.\ \lambda y.\ \lambda z.x\ z\ (y\ z))$ *(S),*
e) $\lambda x.\ \lambda y.\ x\ y$

α-équivalence, variables libres et liées

La relation d'α-équivalence est une relation d'équivalence et c'est aussi une relation de congruence : si e et e' sont α-équivalents alors $\lambda x.\ e$ *et* $\lambda x.\ e'$ le sont aussi, $e\ e_1$ et $e'\ e_1$ le sont aussi, et $e_1\ e$ et $e_1\ e'$ le sont aussi. Quelques exemples et contre-exemples d'*α-équivalence* :

1. $\lambda x.\ \lambda y.\ y$, $\lambda y.\ \lambda x.\ x$, et $\lambda x.\ \lambda x.\ x$ sont α-équivalents.
2. $\lambda x.\ x\ y$ et $\lambda y.\ y\ y$ ne sont pas α-équivalents.

Substitution

On peut remplacer toutes les occurrences d'une variable liée par un λ-terme, mais il faut faire attention que cette opération soit stable par α-équivalence.

Plus précisément, nous noterons $e[e'/x]$ le terme obtenu en remplaçant toutes les occurrences libres de x par e' dans e.

L'opération doit en fait se faire en deux temps :

1. d'abord construire un terme α-équivalent à e où aucune des abstractions n'utilise x ou l'une des variables libres de e'
2. ensuite remplacer toutes les occurrences de x par e'

Les variables libres de $e[e'/x]$ sont alors les variables libres de e et les variables libres de e' .

On peut également décrire récursivement l'opération de substitution par les équations suivantes :

i. $x[e'/x] = e'$,
ii. $y[e'/x] = y$, si $y \neq x$,
iii. $(e1\ e_2)[e'/x] = e_1[e'/x]\ e_2[e'/x]$,
iv. $(\lambda x.\ e)[e'/x] = \lambda x.\ e$,
v. $(\lambda y.\ e)[e'/x] = \lambda y.(e[e\ 0/x])$, *si y n'est pas libre dans e'*,
vi. $(\lambda y.\ e)[e'/x] = \lambda z.((e[z/y])[e'/x])$, *si z n'apparait pas libre dans e et e'*.

La dernière équation s'applique aussi quand les deux précédentes s'appliquent, mais on appliquera celles-ci de préférence quand c'est possible.

Si l'on applique la dernière alors que les précédentes s'appliquent, on obtient simplement un terme α-équivalent

Conversions *α−conversion* :

$$(\lambda x.M) \rightarrow_{\alpha} (\lambda y.M[y/x]) \quad y \text{ non libre dans } M$$

Convention

Nous ne distinguerons pas les termes qui sont identiques après (éventuellement plusieurs) α −conversion.

Conversions ***β−conversion*** :

$$((\lambda x.M)N) \rightarrow_{\beta} M[N/x]) \quad \text{si la substitution est sûre}$$

Réduction

Ces deux conversions sont étendues aux sous-termes :

$$(\lambda x.M) \rightarrow (\lambda x.M') \text{ Si } M \rightarrow M'$$
$$(MN) \rightarrow (M'N) \text{ Si } M \rightarrow M'$$
$$(MN) \rightarrow (MN') \text{ Si } M \rightarrow M'$$

B−réduction

$$\rightarrow^{*}_{B} \cap_{n \in N} \rightarrow^{n}_{B} \text{ représente le calcul sur un } \lambda\text{−terme}$$

Exemple

- $((\lambda x\ .\ \lambda y.\ (xy))b)c$
- $((\lambda x\ .(a(\ \lambda y.\ (xy)))b)c$

Forme normale Définition (Rédex)

- Un rédex d'un terme est une sous-expression de la forme

$$((\lambda x.X)Y)$$

- Un terme est en forme normale s'il ne contient aucun rédex.
- Un terme en forme normale ne peut plus être β−réduit (le calcul est terminé).

Définition (Terme normalisable)

- Un terme E est normalisable s'il existe un terme E' en forme normale telle que

$$E \rightarrow^*_B E'$$

I. $(\lambda x.\ (x\ x))\ (\lambda x.\ (x\ x))$ est-il normalisable ?

II. Donner un exemple de terme normalisable pour lequel il existe une séquence infinie de β−réduction.

Stratégie de réduction Il existe plusieurs façons de réduire un terme.

Théorème (Church-Rosser)
Si $E \rightarrow^*_B E_1$ *et* $E \rightarrow^*_B E_2$*, alors il existe* E' *tel que* $E1 \rightarrow^*_B E'$ *et* $E2 \rightarrow^*_B E'$.
Ce théorème assure l'unicité de la forme normale (exercice).

Il existe une stratégie gagnante :

Définition (Réduction normale)
La réduction normale d'un terme est la stratégie qui applique la β-conversion aux rédex le plus à gauche.

Théorème (Curry)
Si E est normalisable vers une forme normale N alors la réduction normale de E mène aussi à N. Autrement dit, la réduction normale d'un terme normalisable termine toujours.

Nombres naturels
On peut représenter les nombres entiers positifs par les expressions de la forme

$$\lambda f\ x.f(f \cdots (f\ x) \cdots)$$

Par exemple, 0 est le terme $\lambda f\ x.x$, 1 est le terme $\lambda f\ x.\ f\ x$, 2 est le terme $\lambda f\ x.\ f(f\ x)$, et ainsi de suite.
On peut alors représenter les fonctions arithmétiques les plus basiques de la façon suivante :

- **Incrémentation**

$$\lambda x\ f\ z.x\ f(f\ z)$$

- **Addition**

$$\lambda x\ y\ f\ z.\ x\ f(y\ f\ z)$$

- **Multiplication**

$$\lambda x\ y\ f\ z.x\ (y\ f)\ z$$

Dans la suite nous noterons *succ*, plus et *mult* ces fonctions. Dans la littérature ces nombres sont appelés entiers de Church.

Exemple d'application

- **i.** Donner une représentation alternative de *incr*.
- **ii.** Ecrire les nombres *2* et *3*, construire la dérivation correspondant à *2 + 3*, reconnait-on bien le nombre *5* ?
- **iii.** Ecrire la dérivation correspondant à *2 × 3*.
- **iv.** Ecrire la fonction d'exponentiation.

Série d'exercices proposés

Exercice avec solution

1. réduire les termes suivants à des termes bêta-normaux

$$(a) - (\lambda x.xy)(\lambda u.vuu)$$

$$(b) - (\lambda xy.yx)uv$$

$$(c) - \big(\lambda x.x\big(x(yz)\big)x\big)(\lambda u.uv)$$

$$(d) - (\lambda x.xxy)(\lambda y.yz)$$

$$(e) - (\lambda xy.xyy)(\lambda u.uyx)$$

$$(f) - \big(\lambda xyz.xz(yz)\big)\big((\lambda xy.yx)u\big)\big((\lambda xy.yx)v\big)w$$

Solution

$$(e) - (\lambda xy.xyy)(\lambda u.uyx)$$

$$(\lambda xy.xyy)(\lambda u.uyx) \equiv \underline{\big(\lambda x.(\lambda y.xyy)\big)(\lambda u.uyx)}$$

$$\underline{\big(\lambda x.(\lambda y.xyy)\big)(\lambda u.uyx)} \rhd_{1\beta} (\lambda y.xyy)\left[{}^{(\lambda u.uyx)}/_{x}\right]$$

$$\equiv_{\alpha} (\lambda z.xzz)\langle{}^{(\lambda u.uyx)}/_{x}\rangle$$

$$\equiv \big(\lambda z.\underline{(\lambda u.uyx)zz}\big)$$

$$\rhd_{1\beta} (\lambda z.(uyx)\left[{}^{z}/_{u}\right]z)$$

$$\equiv \lambda z.zyxz$$

$$(\lambda xy.xyy)(\lambda u.uyx) \equiv_\alpha \underline{(\lambda z.(\lambda t.ztt))(\lambda u.uyx)}$$

$$\underline{(\lambda z.(\lambda t.ztt))(\lambda u.uyx)} \rhd_{1\beta} (\lambda t.ztt)\langle {}^{(\lambda u.uyx)}/_{z}\rangle$$

$$\equiv \left(\lambda t.\underline{(\lambda u.uyx)tt}\right)$$

$$\rhd_{1\beta} (\lambda t.(uyx)\langle {}^{t}/_{u}\rangle z)$$

$$\equiv \lambda t.tyxt$$

$$(f) - (\lambda xyz.xz(yz))(\underbrace{(\lambda xy.yx)}u)(\underbrace{(\lambda xy.yx)}v)w$$

$$((\lambda xy.yx)u) \equiv \underline{(\lambda x.(\lambda y.yx))u} \quad ((\lambda xy.yx)v) \rhd_{1\beta} \lambda y.yv$$

$$\rhd_{1\beta} (\lambda y.yx)\left[{}^{u}/_{x}\right] \equiv \lambda y.yu$$

$$\rhd_\beta (\lambda xyz.xz(yz))(\lambda y.yu)(\lambda y.yv)w$$

$$\equiv_\alpha \left(\lambda t.\left(\lambda m.(\lambda n.tn(mn))\right)\right)(\lambda y.yu)(\lambda y.yv)w$$

$$\rhd_{1\beta} \overline{\left(\left(\lambda m.(\lambda n.tn(mn))\right)\left[{}^{(\lambda y.yu)}/_{t}\right]\right)}(\lambda y.yv)w$$

$$\rhd_{1\beta} \left(\left(\lambda m. \left(\lambda n. tn(mn) \right) \right) \left[{(\lambda y. yu)}/{t} \right] \right) (\lambda y. yv)w$$

$$\equiv \underline{\left(\lambda m. \left(\lambda n. (\lambda y. yu)n(mn) \right) \right) (\lambda y. yv)w}$$

$$\rhd_{1\beta} \left(\left(\lambda n. (\lambda y. yu)n(mn) \right) \left[{(\lambda y. yv)}/{m} \right] \right) w$$

$$\equiv \underline{\left(\lambda n. (\lambda y. yu)n\left((\lambda y. yv)n \right) \right) w}$$

$$\rhd_{1\beta} \left((\lambda y. yu)n\left((\lambda y. yv)n \right) \right) \left[{w}/{n} \right]$$

$$\equiv \underline{(\lambda y. yu)w}\left((\lambda y. yv)w \right) \rhd_{1\beta} wu\left((\lambda y. yv)w \right)$$

$$\rhd_{1\beta} wu(wv)$$

Exercice 1.

Le terme suivant est il syntaxiquement correct ? Si oui, décomposez-le en sous-termes en délimitant toutes les abstractions et applications.

- $xy\ .\lambda y.x$
- $\lambda x.xyy$
- $\lambda x.y\ xx$
- $\lambda zx.\ x.\lambda y\ z$
- $\lambda x.xy\lambda t.tx$
- $\lambda x.tx$

Exercice 2.

Soit *M* un terme du _-calcul ; réduire autant que possible les expressions suivantes :

add *M* **0**, **add 0** *M*, **mul** *M* **0**, **mul 0** *M*, **mul** *M* **1**, **mul 1** *M*, et **exp** *M* **0**.

Exercice 3.

1. réduire (**true** *M*) et (**false** *M*), où *M* désigne un terme quelconque ;
2. donner des définitions pour les opérateurs logiques **and**, **or** et **not**. *Conseil* : ces définitions commencent par **if** .

Exercice 4.

Ecrire un terme qui code le test **iszero**, de telle sorte que **iszero 0 true** et **iszero n false** si **n** est un entier de Church différent de **0**

Exercice 5.

On code un *couple* par le terme **pair** = _ *x y f* . *f x y*

Donner les définitions de **first** et **second** de telle sorte que, pour tous termes *M*, *N* on ait les réductions :

first (**pair** *M N*) *M* , **second** (**pair** *M N*) *N*.

Exercice 6. (Examen 2015)

Réduire et donner la forme normale des termes suivants :

1. ((λx.(x x)) (λz.z))
2. ((λxy.y) f g)
3. ((λxy.(x (x (x y)))) (λx.z))

Chapitre 7 Complexité & NP Complet

1. Résumé du cours

La complexité en espace correspond à l'espace mémoire (nombre d'octets) nécessaire à l'exécution sur une donnée de taille N.
La complexité en temps correspond au temps d'exécution (nombre d'opérations élémentaires) de l'algorithme sur une donnée de taille N.

- Le principe de la complexité est d'avoir une mesure pour évaluer les performances d'un algorithme (temps d'exécution ou espace mémoire)
- Le but est de comparer les algorithmes résolvant un même problème

2. Ce qu'il faut retenir

A un problème donné, il existe plusieurs solutions algorithmiques.
Il faut les comparer par rapport à leurs complexités.
Il faut s'assurer que l'algorithme proposé est le meilleur possible pour le problème considéré.

En réalité, chaque problème a sa propre complexité (connue ou non). Dès qu'un algorithme à la même complexité, il est optimal.
Plus formellement : soient *A* un algorithme, d la donnée de cet algorithme, D_n l'ensemble de toutes les données possibles, de taille *n* :

Coût $_{espace}$ *(A, d) (en octets)* et *coût* $_{temps}$ *(A, d)* (en nombre d'opérations élémentaires) de l'exécution de *A* sur *d*

Dans ce chapitre, On s'intéresse à trois types de coût :

- *Le coût minimum* : il correspond au meilleur des cas : $Min_n(A) = min_{d \in Dn} \{coût\ (A, d)\}$

- *Le coût maximum* : il correspond au pire des cas : $Max_n(A) = max_{d \in Dn} \{coût\ (A, d)\}$

Le coût moyen : $Moy_n(A) = \Sigma_{d \in Dn} P(d) * \{coût\ (A, d)\}$ Ou *P(d)* est la probabilité d'apparition de la donnée *d*. Si les données sont équiprobables alors $P(d) = 1/(Card[D_n])$

Notations

Soient deux fonctions f et g $N \rightarrow N$

On dit que $f=O(g)$ « f est dominée par g » ⇔ A partir d'un rang n_0 $f(n) \leq Cste.g(n)$

On dit que $f=\Theta(g)$ « f et g sont asymptotiquement équivalentes » ⇔ $f=O(g)$ et $g=O(f)$
En pratique, on exprime donc les complexités sous la forme $\Theta(g)$ avec g très simple.

Exemples

- $\Theta(1)$: complexité constante (exemple : la somme des entiers de 1 à n)
- $\Theta(log(n))$: logarithmique (exemple : recherche dichotomique)
- $\Theta(n)$: complexité linéaire (exemple : recherche séquentielle)
- $\Theta(n*log(n))$: complexité sub-quadratique (exemple : tri rapide)
- $\Theta(n^2)$: complexité quadratique (exemple : tris simples)
- $\Theta(n^k)$: complexité polynomiale
- $\Theta(2^n)$: c (exemple : Tours de Hanoi)

On fera donc par exemple : $3n^3+6n^2+17n+3 \rightarrow \Theta(n^3)$

Influence de la complexité sur l'exécution : Prenons l'exemple d'une machine qui fait une opération élémentaire en 1µs (soit 10^6 opérations par seconde)

Taille de n \ Complexité	*1*	*log(n)*	*n*	*n*log(n)*	*n²*	*n³*	*2ⁿ*
100	*1µs*	*7 µs*	*100 µs*	*0.7 ms*	*10 ms*	*1 s*	*4.10^{16} années*
10000	*1 µs*	*14 µs*	*10 ms*	*0.14 s*	*1,5 min*	*11.5 jours*	*infini*
10^6	*1 µs*	*20 µs*	*1s*	*20s*	*11.5 jours*	*32.10^6 années*	*infini*

Détermination de la complexité

Dans l'espace

C'est en général facile, surtout pour les données statiques : la taille est connue en fonction du type.
Pour les ordres de grandeurs, seuls les tableaux comptent.
La seule difficulté apparaît dans les allocations dynamiques (mémoire demandée en cours d'exécution) ou dans les algorithmes récursifs (il faut compter le nombre d'appels pour avoir une idée de la taille.

Dans le temps

- Enchaînements séquentiels : L'ordre de grandeur de la séquence correspond à la valeur maximale.
- Séquences d'actions élémentaires : $\Theta(1)$
- S'il y a un appel de fonction, il faut inclure la complexité de cette fonction dans la complexité de la séquence.
- Alternative (Si…alors…sinon) : On prend toujours la complexité maximale. Sinon on peut aussi faire une complexité moyenne mais il faut avoir des informations sur le résultat moyen de la condition.
- Boucles : Complexité = (Complexité du corps de boucle) *x* (Nombre de tours).
- Algorithme récursif : Complexité = (Complexité de la fonction) *x* (nombre d'appels).

Structures de données et algorithmes en pratique

La résolution de problèmes algorithmiques requiert presque toujours la combinaison de structures de données et d'algorithmes sophistiqués pour la gestion et la recherche dans ces structures.

D'autant plus vrai qu'on à traiter des volumes de données importants.
Quelques exemples de problèmes réels :

1. Routage dans les réseaux informatiques
2. Moteurs de recherche
3. Alignement de séquences ADN en bio-informatique

Exemples : Les algorithmes de tris

Les algorithmes de tris ont deux types d'opérations fondamentales : les comparaisons et les transferts.

On peut distinguer deux types de tris :

- Les tris internes : en mémoire centrale :
    - Tri d'un tableau
    - Tri d'un tableau sur lui-même
- Les tris externes : tri de fichiers

Exercice proposé sur la Classes de complexité P et NP

Exercice 1.

La méthode naïve pour chercher le plus petit diviseur d non trivial d'un entier p impair consiste à tester tous les entiers d impairs inférieurs à $p1/2$, jusqu'à trouver un diviseur (en cas d'échec, on en déduit que p est premier)

Exprimer la complexité de cet algorithme en utilisant la notation "grand-O", d'abord en fonction de p, ensuite en fonction de la *taille* de

Exercice 2.

Montrer qu'une expression $O(\ln n)$ est aussi $O(\log_2 n)$, et vice-versa.
Les notations $O(e^n)$ et $O(2^n)$ sont-elles équivalentes ?

Exercice 3.

1. Calculer la table de vérité de la formule : $\Gamma = (x \wedge \neg y) \vee (y \wedge \neg z) \vee z$
2. Trouver une formule £ de la forme $x' \wedge y' \wedge z'$, avec $x' = x$ ou $\neg x$, etc. telle que £ *et* Γ soit une tautologie
3. Transformer $(x \wedge \neg y) \vee (y \wedge \neg z)$ en forme normale conjonctive

Exercice 4.

Transformer la formule $(x_1 \vee y_1) \wedge (x_2 \vee y_2) .. \wedge .. (x_n \vee y_n)$ en forme normale disjonctive

Exercice 5.

1. Exprimer l'égalité $y = \neg x$ sous forme de conjonction de clauses
2. Même question pour $z = x \wedge y$
3. Même question pour $z = x \vee y$

Exercice 6.

1. Exprimer l'implication $x \Rightarrow (\neg y \wedge \neg z)$ sous forme de conjonction de clauses.
2. En déduire une formule en forme normale conjonctive, qui exprime que l'une des trois variables x, y, z vaut 1, et une seulement.
3. Généraliser au cas de n variables $x_1, x_2, \ldots, x_n$.
4. Prouver directement, c'est-à-dire sans utiliser la question 1, que la formule obtenue est correcte

Exercice 7.

Prouver que la classe P est close par union, concaténation et complément

Exercice 8.

Prouver que la classe NP est close par union et concaténation

Exercice 9.

Prouver que si NP n'est pas égal à son complément, alors P≠NP

Exercice 10.

Prouver que si P = NP alors tout langage $A \in P$ sauf $A = \phi$ et $A = \sum^*$ sont NP-complets

Référence :

1. *Logique mathématique, tome 2 : Fonctions récursives, théorème de Gödel, théorie des ensembles, théorie des modèles*, *René Cori, Daniel Lascar. Éd. Dunod, Paris, 2003.*
2. *Fonctions récursives primitives (définitions et résultats sous forme d'exercices corrigés) par Bruno Salvy (Ecole Polytechnique) : http://www.enseignement.polytechnique.fr/informatique/INF423/uploads/Main/pc4.pdf*
   Corrigé : http://www.enseignement.polytechnique.fr/informatique/INF423/uploads/Main/pc4-correction.pdf
3. *LTF Gamut [1992] : Logic, Language and Meaning, vol 1 : Introduction to logic, University of Chicago Press, Chicago, en particulier les chapitres 1 et 2, pp. 1-64. [Ouvrage remarquable par sa clarté didactique et par la richesse de ses exemples. Les méthodes de preuves introduites (déduction naturelle) sont cependant différentes de celles présentées ici.]*

4. *R. Smullyan [1968] : First Order Logic, Dover Publications, New York, en particulier la première partie, pp. 3-40*

5. *R. Lassaigne et M. de Rougemont. Logique et fondements de l'informatique. Hermes, 1993.*

6. *M.-D. Popelard et D. Vernant. Éléments de logique. Mémo Seuil, 1998. [3] S. Russel et P. Norvig. Intelligence artificielle (2e édition). Pearson Education, 2006.*

Index

(A_g, r, A_d), voir arbre binaire	(V, E), voir graphe
(q, u, v), voir configuration d'une machine de Turing	union de deux ensembles notation, voir ∪
., voir concaténation	Ac
, voir complémentaire	F(G/p), voir substitution
L(M), voir langage accepté par une machine	de Turing
⇔, voir double implication	⇒, voir définition inductive / différentes
notations d'une, voir implication	Σ, voir alphabet
Σ, voir ensemble des mots sur un alphabet	∗
R, voir décidable	∩, voir intersection de deux ensembles
∪, voir union de deux ensembles	, voir mot vide
≡, voir équivalence entre formules,	voir équivalence entre problèmes
∃, voir quantificateur	∀, voir quantificateur
≤m, voir réduction	|w|, voir longueur d'un mot
≤, voir réduction	P(E), voir parties d'un ensemble
|=, voir conséquence sémantique	¬, voir négation
|=, voir conséquence sémantique	⊂, voir partie d'un ensemble
×, voir produit cartésien de deux ensembles	├, voir démonstration, voir relation successeur entre configurations d'une machines de Turing
notation , voir HALTING − PROBLEM	∧, voir conjonction
RE-complétude, voir RE-complet	∨, voir disjonction

4.2. **Tableau de vérité.** Nous présentons ces définitions en forme de tableaux de vérité, où V:=vrai, et F:=faux.

Pour la négation:

p	$\neg p$
V	F
F	V

Pour les autres définitions:

p	q	$p \vee q$	$p \oplus q$	$p \wedge q$	$p \to q$	$p \leftrightarrow q$
V	V	V	F	V	V	V
V	F	V	V	F	F	F
F	V	V	V	F	V	F
F	F	F	F	F	V	V

4.3. **Propositions logiques composées.** Commençons maintenant à combiner. Si p, q et r sont trois propositions logiques, alors

$$P := ((\neg q) \wedge (p \vee r)) \to p$$

est aussi une proposition logique. La valeur de vérité de P dépend des valeurs de vérité de p, q et r. Comment exactement montre le tableau suivant (qui montre aussi des calculs de vérité intermédiaires) :

p	q	r	$\neg q$	$p \vee r$	$(\neg q) \wedge (p \vee r)$	$((\neg q) \wedge (p \vee r)) \to p$
V	V	V	F	V	F	V
V	V	F	F	V	F	V
V	F	V	V	V	V	V
V	F	F	V	V	V	V
F	V	V	F	V	F	V
F	V	F	F	F	F	V
F	F	V	V	V	V	F
F	F	F	V	F	F	V

En regardant la dernière colonne on se rend compte que P est faux si et seulement si p est faux et q est faux et r est vraie, ou en formule que

$$(\neg p) \wedge ((\neg q) \wedge r))$$

est vraie.

Ou que P est vraie si et seulement si

$$\neg ((\neg p) \wedge ((\neg q) \wedge r))$$

est vraie.

Posons $Q := \neg ((\neg p) \wedge ((\neg q) \wedge r))$ et calculons de nouveau ses valeurs de vérité :

p	q	r	$\neg p$	$\neg q$	$(\neg q) \wedge r)$	$((\neg p) \wedge ((\neg q) \wedge r))$	Q
V	V	V	F	F	F	F	V
V	V	F	F	F	F	F	V
V	F	V	F	V	V	F	V
V	F	F	F	V	F	F	V
F	V	V	V	F	F	F	V
F	V	F	V	F	F	F	V
F	F	V	V	V	V	V	F
F	F	F	V	V	F	F	V

Considérons aussi la proposition $R = p \vee (q \vee (\neg r)$. On peut aussi calculer son tableau.

p	q	r	P	Q	R
V	V	V	V	V	V
V	V	F	V	V	V
V	F	V	V	V	V
V	F	F	V	V	V
F	V	V	V	V	V
F	V	F	V	V	V
F	F	V	F	F	F
F	F	F	V	V	V

Conclusion : les propositions composées P, Q et R ont la même valeur de vérité, n'importe les valeurs de p, q,r. On dit P, Q et R sont des propositions composées (ou formules logiques) *logiquement équivalentes.*

Calculons aussi les vérités de la proposition $P \leftrightarrow Q$:

p	q	r	P	Q	$P \leftrightarrow Q$
V	V	V	V	V	V
V	V	F	V	V	V
V	F	V	V	V	V
V	F	F	V	V	V
F	V	V	V	V	V
F	V	F	V	V	V
F	F	V	F	F	V
F	F	F	V	V	V

On voit que la proposition logique composée $P \leftrightarrow Q$ est toujours vraie, pour tous les valeurs de vérité de p, q et r. On dit $P \leftrightarrow Q$ est une *tautologie*, notation $P \Leftrightarrow Q$.

4.4. **Tautologie et contradiction.** Nous généralisons.

Définition 4.2. *Une proposition logique composée (ou une formule logique) qui est toujours vraie, quelles que soient les valeurs de vérité des propositions qui la composent, est appelée une tautologie. Notation : $P \Leftrightarrow V$.*

Par exemple: $p \vee (\neg p) \Leftrightarrow V$ et $((p \wedge q) \rightarrow (p \vee q)) \Leftrightarrow V$.

Définition 4.3. *Une proposition logique composée (ou une formule logique) qui est toujours fausse, quelles que soient les valeurs de vérité des propositions qui la composent, est appelée une contradiction. Notation :* $P \Leftrightarrow F$

Par exemple : $p \wedge (\neg p) \Leftrightarrow F$.

Définition 4.4. *Deux formules logiques* P *et* Q *sont appelées* logiquement équivalentes *si la proposition logique* $P \leftrightarrow Q$ *est une tautologie. Notation :* $P \Leftrightarrow Q$.

Par exemple: $(p \leftrightarrow q) \Leftrightarrow [(p \to q) \wedge (q \to p)]$.

Définition 4.5. *Si* $P \to Q$ *est une tautologie, où* P *et* Q *sont deux formules logiques, on dit que* $P \to Q$ *est une* règle d'inférence. *Notation* $P \Rightarrow Q$.

Par exemple: $(p \wedge q) \Rightarrow q$.

4.5. **Propositions logiquement équivalents à l'implication.** Considérons la proposition suivante.

Proposition 4.1. *Les trois formules logiques "*$p \to q$*", "*$(\neg q) \to (\neg p)$*" et "*$(\neg p) \vee q$*" sont logiquement équivalentes. Ou*

$$(p \to q) \Leftrightarrow ((\neg q) \to (\neg p)) \Leftrightarrow ((\neg p) \vee q).$$

Proof. Nous vérifions par un tableau de vérité.

p	q	$p \to q$	$\neg q$	$\neg p$	$(\neg q) \to (\neg p)$	$(\neg p \vee q)$
V	V	V	F	F	V	V
V	F	F	V	F	F	F
F	V	V	F	V	V	V
F	F	V	V	V	V	V

Les trois colonnes correspondantes sont identique. Donc la proposition est vraie. □

4.6. **Formules logiquement équivalentes.** Donnons une petite reformulation. On pourrait voir

$$P = P(p, q, r),\ Q = Q(p, q, r),\ R = R(p, q, r)$$

comme trois formules ou comme trois fonctions qui dépendent des propositions logiques p, q, r. Par exemple

$$P : \{\text{propositions logiques}\}^3 \to \{\text{propositions logiques}\}$$

où le triple de propositions (p, q, r) est envoyé vers la proposition $P = P(p, q, r)$.

Par définition: P et Q sont logiquement équivalentes si les fonctions composées avec la fonction "vérité" deviennent identiques:

$$\text{"vérité"} \circ P = \text{"vérité"} \circ Q.$$

Si A et B sont deux ensembles et $b \in B$ un élément fixé, on peut définir la *fonction constante*, souvent aussi noté b, comme la fonction qui associe à chaque élément $a \in A$ ce même élément b (ou $b(a) = b$). Dans le même sens nous considérons V comme la proposition composée (ou formule logique) qui a la valeur de vérité V n'importe les valeurs de p, q, r. Et de même pour F.

4.7. **Équivalences logiques de base.** Une première liste d'équivalences logiques utilisées tout le temps :

Théorème 4.1. *Voici des équivalences logiques :*

$$
\begin{aligned}
p \wedge V &\Leftrightarrow p \ \textit{(Identité)} \\
p \vee F &\Leftrightarrow p \ \textit{(Identité)} \\
p \vee V &\Leftrightarrow V \ \textit{(Domination)} \\
p \wedge F &\Leftrightarrow F \ \textit{(Domination)} \\
p \vee p &\Leftrightarrow p \ \textit{(Idempotence)} \\
p \wedge p &\Leftrightarrow p \ \textit{(Idempotence)} \\
\neg(\neg p) &\Leftrightarrow p \ \textit{(Loi de la double négation)} \\
p \wedge q &\Leftrightarrow q \wedge p \ \textit{(Commutativité)} \\
p \vee q &\Leftrightarrow q \vee p \ \textit{(Commutativité)} \\
((p \vee q) \vee r) &\Leftrightarrow (p \vee (q \vee r)) \ \textit{(Associativité)} \\
((p \wedge q) \wedge r) &\Leftrightarrow (p \wedge (q \wedge r)) \ \textit{(Associativité)} \\
(p \vee (q \wedge r)) &\Leftrightarrow ((p \vee q) \wedge (p \vee r))) \ \textit{(Distributivité)} \\
(p \wedge (q \vee r)) &\Leftrightarrow ((p \wedge q) \vee (p \wedge r))) \ \textit{(Distributivité)} \\
\neg(p \wedge q) &\Leftrightarrow ((\neg p) \vee (\neg q)) \ \textit{(Loi de De Morgan)} \\
\neg(p \vee q) &\Leftrightarrow ((\neg p) \wedge (\neg q) \ \textit{(Loi de De Morgan)}
\end{aligned}
$$

Proof. La plupart des preuves est facile: il faut soi-même se convaincre! Nous donnons deux exemples de preuve par tableau :

Si $P := p \vee (q \wedge r)$ et $Q := (p \vee q) \wedge (p \vee r)$ on a

p	q	r	$q \wedge r$	P	$p \vee q$	$p \vee r$	Q
V	V	V	V	V	V	V	V
V	V	F	F	V	V	V	V
V	F	V	F	V	V	V	V
V	F	F	F	V	V	V	V
F	V	V	V	V	V	V	V
F	V	F	F	F	V	F	F
F	F	V	F	F	F	V	F
F	F	F	F	F	F	F	F

Effectivement $P \Leftrightarrow Q$ (un des deux lois de la distributivité).

Montrons $\neg(p \wedge q) \Leftrightarrow ((\neg p) \vee (\neg q))$ (Loi de De Morgan). Soit $P = \neg(p \wedge q)$ et $Q = ((\neg p) \vee (\neg q))$. On a

p	q	$p \wedge q$	P	$\neg p$	$\neg q$	Q
V	V	V	F	F	F	F
V	F	F	V	F	V	V
F	V	F	V	V	F	V
F	F	F	V	V	V	V

Effectivement $P \Leftrightarrow Q$. □

4.8. **Preuve d'equivalences logiques par de l'algèbre.** Remarquons tout d'abord que si P, Q et R sont trois formules logiques (ou propositions composées) et $P \Leftrightarrow Q$ et $Q \Leftrightarrow R$ alors aussi $P \Leftrightarrow R$. (Et si $P \Leftrightarrow Q$ alors aussi $Q \Leftrightarrow P$ (c.-à.d., l'équivalence logique est symmétrique), et on a toujours $P \Leftrightarrow P$.)

Donc on peut enchaîner des équivalences logiques pour obtenir d'autres.

Nous avons déjà un certain nombre d'équivalence montrées, que nous pouvons utiliser. À la place d'utiliser un tableau pour vérifier une équivalence logique, on peut aussi utiliser les petites règles déjà montrées, comme en algèbre.

Donnons un exemple. La proposition soi-même est sans importance.

Proposition 4.2. *Soit* $P := ((\neg q) \wedge (p \vee r)) \to p$ *et* $Q := (p \vee (q \vee (\neg r)))$. *On a* $P \Leftrightarrow Q$.

Proof. Nous allons utiliser une suite d'équivalences logiques (avec les raisons)

$$\begin{aligned}
P &\Leftrightarrow (\neg((\neg q) \wedge (p \vee r))) \vee p \text{ (Car } (p \to q) \Leftrightarrow ((\neg p) \vee q)) \\
&\Leftrightarrow (\neg(\neg q) \vee \neg(p \vee r))) \vee p \text{ (Selon De Morgan)} \\
&\Leftrightarrow (q \vee \neg(p \vee r))) \vee p \text{ (Double négation)} \\
&\Leftrightarrow (q \vee ((\neg p) \wedge (\neg r))) \vee p \text{ (Selon De Morgan)} \\
&\Leftrightarrow q \vee (p \vee ((\neg p) \wedge (\neg r))) \text{ (Assoc. et comm. pour } \vee) \\
&\Leftrightarrow q \vee ((p \vee (\neg p)) \wedge (p \vee (\neg r))) \text{ (Distrib.)} \\
&\Leftrightarrow q \vee (V \wedge (p \vee (\neg r))) \text{ (Selon } (p \vee (\neg p)) \Leftrightarrow V) \\
&\Leftrightarrow q \vee (p \vee (\neg r)) \text{ (Selon comm. et } (p \wedge V) \Leftrightarrow p) \\
&\Leftrightarrow Q \text{ (Assoc. et commut.)}
\end{aligned}$$

Et voilà ! □

Chaque ligne est claire. Mais trouver la chaîne n'est pas si évidente.

Remarque. On a vu beaucoup de ()'s dans cette preuve. Parce que $p \vee (q \vee r) = (p \vee q) \vee r$, écrire

$$p \vee q \vee r$$

n'est pas ambigu. On supprime des ()'s, qui sont dans ce cas superflus.

Ainsi pour

$$p \wedge q \wedge r$$

On pose la convention que "$\neg$" a une *plus haute priorité* que $\vee$ ou $\wedge$ ou $\to$ ou ...

Donc $\neg p \vee q$ veut dire $(\neg p) \vee q$ (et pas $\neg(p \vee q)$, ce qui est vraiment différent).

Et $\neg p \to q$ veut dire $(\neg p) \to q$.

Mais c'est ambigu d'écrire "$p \wedge q \vee r$" et "$p \wedge q \to r$"; on n'a pas le droit d'écrire ça (il faut ajouter des ()'s pour clarifier ce qu'on veut vraiment). Nous ne posons pas de priorités entres les autres opérations. **Vous aussi êtes obligés d'évitez des ambiguïtés**, en utilisant les ()'s et [ ],s. Et à chaque symbol "(" il faut correspondre un symbol ")".

4.9. **Règles d'inference de base.** Voici quelques règles d'inférence[6] utilisées tout le temps aussi.

Théorème 4.2. *Les suivants sont des équivalences logiques.*

$$\begin{aligned}
p &\Rightarrow (p \vee q) \ \textit{(Addition)} \\
p \wedge q &\Rightarrow p \ \textit{(Simplification)} \\
[p \wedge (p \to q)] &\Rightarrow p \ \textit{(Modus ponens)} \\
[\neg q \wedge (p \to q)] &\Rightarrow q \wedge p \ \textit{(Modus tollens)} \\
[(p \to q) \wedge (q \to r)] &\Rightarrow (p \to r) \ \textit{(Syllogisme par hypothèse)} \\
[(p \vee q) \wedge \neg q] &\Rightarrow p \ \textit{(Syllogisme disjonctif)}
\end{aligned}$$

Proof. Aussi facile à montrer avec un tableau de vérité. Par exemple *Modus ponens*:

p	q	$p \to q$	$p \wedge (p \to q)$	$(p \wedge (p \to q)) \to p$
V	V	V	V	V
V	F	F	F	V
F	V	V	F	V
F	F	V	F	V

La dernière colonne contient seulement des V, donc correspond à une tautologie. □

Si $P \Rightarrow Q$ et $Q \Rightarrow R$ alors aussi $P \Rightarrow R$. (Mais $P \Rightarrow Q$ n'implique pas en général que $Q \Rightarrow P$).

Et $P \Leftrightarrow Q$ si et seulement si $P \Rightarrow Q$ et $Q \to P$.

Donc on peut composer des règles d'inférences, et obtenir d'autre règles d'inférence.

Par exemple.

Lemme 4.1. *Une autre règle d'inférence :*

$$(q \wedge (\neg p \to \neg q)) \Rightarrow p.$$

Proof. On combine $(\neg p \to \neg q) \Leftrightarrow (q \to p)$ avec modus ponens

$$(q \wedge (\neg p \to \neg q)) \Leftrightarrow (q \wedge (q \to p)) \Rightarrow p.$$

□

Cette règle d'inférence est utilisée dans une preuve par l'absurde, comme nous verrons plus tard.

[6] Voir [R, §3.1]

4.10. **Fonctions propositionnelles et quantifications.** Il y a des énoncés comme "$n > 2$" qui sont presque des propositions logiques mais qui contiennent des indéterminés (comme "n" dans l'exemple). Du moment qu'on précise ces indéterminés elles deviennent des propositions logiques. C'est donc vraiment une famille de propositions logiques.

Par exemple $p(n) :=$ "$n > 2$". C'est une famille de propositions logiques, dépendant de la variable n qui varie dans l'ensemble $\mathbb{N}$. Ici $p(n)$ est fausse si $n = 0, 1$ ou 2 et vraie sinon (pour $n \in \mathbb{N}$).

Soit U un ensemble. Une fonction

$$p : U \to \{\text{propositions logiques}\}$$

qui associe à chaque $u \in U$ la proposition logique

$$p(u)$$

s'appelle *fonction propositionnelle* avec U comme univers du discours de la variable u.

Exemple: Soit U l'ensemble des hommes qui habitent Montréal et $p(X) :=$ " X peut patiner".

Soit $P(u)$ une fonction propositionnelle avec U comme univers de discours de la variable u.

On définit deux nouvelles propositions logiques vraies, les deux *quantifications*.

On écrit

$$\forall u\ P(u) \text{ (ou } \forall u : P(u))$$

pour la proposition logique:

"$P(u)$ est vraie pour tous les valeurs de u sur son univers de discours", ou,

"pour chaque valeur u dans l'univers de discours la proposition $P(u)$ est vraie".

(On dit parfois: la quantification universelle.)

Dans l'exemple:

$\forall X\ P(X)$ est une traduction logique de "Tous les hommes qui habitent Montréal peuvent patiner."

Et on écrit

$$\exists u\ P(u) \text{ (ou } \exists u : P(u))$$

pour la proposition logique:

"$P(u)$ est vraie pour au moins une des valeurs de u choisi dans son univers de discours", ou

"il existe au une valeur u dans l'univers de discours telle que la proposition $P(u)$ est vraie".

(On dit: parfois: la quantification existentielle.)

Dans l'exemple: $\exists X\ P(X)$ est une traduction logique de "Au moins un homme qui habite Montréal peut patiner."

Exemple: Si $p(n) :=$ "$n > 2$" avec univers du discours $\mathbb{N}$. alors "$\forall n\ n > 2$" est fausse (car au moins $p(1) =$ "$1 > 2$" n'est pas vraie); et "$\exists n\ n > 2$" est vraie (car au moins $p(3) =$ "$3 > 2$" est vraie).

On remarque:

$\forall u\ p(u)$ est vraie si $p(u)$ est vraie pour chaque u;

$\forall u\ p(u)$ est fausse s'il existe au moins un u tel que $p(u)$ est fausse. Traduction

$$\neg(\forall u\ p(u)) \Leftrightarrow \exists u\ (\neg p(u))$$

$\exists u\ p(u)$ est vraie s'il existe un u tel que $p(u)$ est vraie; $\exists u\ p(u)$ est fausse si pour chaque u on a que $p(u)$ est fausse. Traduction:

$$\neg(\exists u\ p(u)) \Leftrightarrow \forall u\ (\neg p(u))$$

On formule ses résultats en forme d'une proposition.

Proposition 4.3. *Soit $P(u)$ une fonction propositionnelle avec U comme univers de discours de la variable u. Alors*

$$\neg(\forall u\ p(u)) \Leftrightarrow \exists u\ (\neg p(u))$$

et

$$\neg(\exists u\ p(u)) \Leftrightarrow \forall u\ (\neg p(u)).$$

4.11. **Utilisation dans les arguments et preuves mathématiques.** On utilise les règles d'inférence dans les arguments et dans les preuves. Typiquement (ici p et q sont certains propositions logiques explicites) :

Un petit argument comme le suivant est logiquement correct (par modus ponens) : *"On a p et aussi que p implique q, donc on a aussi q"*, ce qui est une version courte pour: *"La proposition logique p est vraie et l'implication $p \to q$ est vraie, alors automatiquement la proposition logique q est aussi vraie"*.

Remarque. Si on dit en mathématique "on montre p", ça veut dire "on donne des arguments pour montrer que la proposition logique p est vraie". C'est plus court.

Ou un argument comme: *"p implique q et q implique r, alors nécessairement p implique r aussi"*, ce qui est la version courte de *"Les implications $p \to q$ et $q \to r$ sont vraies alors nécessairement l'implication $p \to r$ est aussi vraie"* (par Syllogisme par hypothèse).

Ou trivialement : *"On sait p et q, donc en particulier p"*, ce qui courte pour : *"On sait que les propositions logiques p et q sont vraie, donc p est vraie"* (par Simplification).

Ou: *"On sait p ou q, mais q est faux; donc nécessairement p est vraie,"* est courte pour: *On sait $p \vee q$ est vraie, mais que la proposition q est fausse; alors la proposition logique p est vraie.*

Tout ça devrait paraître naturel.

Exercice 4.1. (i) Mais comparer maintenant les deux "arguments" semblables suivants.

"On sait que p implique q, mais p est faux; alors q est aussi faux" (courte pour : *On sait que l'implication $p \to q$ est vraie et que la proposition p est fausse; alors la proposition q est fausse*)

et

"On sait que p implique q, mais q est faux; alors p est aussi faux" (courte pour : *On sait que l'implication $p \to q$ est vraie et que la proposition q est fausse; alors la proposition p est fausse*).

La différence semble être petite, mais le premier argument est logiquement **invalide** (car basée sur une contre-vérité) et le deuxième argument est logiquement valide.

Trouver pourquoi.

(ii) Est-ce que l'argument suivant est valide:

"On a q et que q est impliqué par p; alors p.

La (contre-) vérité sous-entendue ?

(iii) Est-ce que vous acceptez : *On sait que q est vraie, alors l'implication $p \to q$ est vraie* ?

(iv) Et : *On sait que p est faux, alors l'implication $p \to q$ est vraie* ?

(v) Et : *On sait que l'implication $p \to q$ est faux, alors q est vraie* ?

4.12. **Preuve par contradiction.** Soit p une proposition logique explicite, qu'on veut montrer vraie. Une méthode est de supposer le contraire est vraie, alors que p est faux. Puis d'utiliser cet hypothèse, et des théorèmes déjà montrés pour obtenir une contradiction ou une absurdité. On conclut que p est vraie.

Encore une fois, comme un modèle.

Proposition 4.4. p

Structure d'une preuve par l'absurde typique. Montrons p par une preuve par l'absurde. Supposons par contre que p soit fausse, c.-à-d que $\neg p$ vraie.

Puis en utilisant cet hypothèse $\neg p$ (et les théorèmes déjà montrés) on montre qu'une certain proposition logique auxiliaire, disons q, est aussi fausse (donc $\neg q$ est vrai). Puis on remarque que c'était déjà connu (ou on avait déjà montré sans utiliser l'hypothèse que q est faux !) que q est vraie. C'est absurde (ou une contradiction).

On conclut que p est vraie. □

Remarque. C'est quoi le principe logique utilisé? Dans la preuve on montre en fait que l'implication $(\neg p) \to (\neg q)$ est vraie et aussi que q est vraie. Par l'inférence $q \wedge (\neg p \to \neg q) \Rightarrow p$ (lemme 4.1) on conclut p est vraie. Ce qu'on voulait montrer en effet. Par exemple, souvent q est une proposition logique qui est évidemment vrai comme $q =$ "$3 > 2$".

4.13. **Preuves typique d'une implication.** En mathématiques il faut souvent montrer des implications $p \to q$, où p et q sont deux propositions logiques. Il y a trois versions. Voir par exemple prop.3.2[7].

Soient p et q deux propositions logiques explicites en mathématiques, et on veut montrer que p implique q.

Théorème 4.3. $p \to q$

Structure d'une preuve directe typique. Si p est fausse, l'implication est automatiquement vraie, donc il n'y a rien à montrer dans ce cas. (Cette phrase est souvent omise).

Supposons p est vraie. Puis (avec cette hypothèse et avec de l'aide des théorèmes déjà montrés), on montre que q sera aussi vraie.

On aura montré que $p \to q$ est vraie. □

Cette version est claire, j'espère. Une autre version:

Structure d'une preuve indirecte typique. Il suffit de montrer sa contraposée $(\neg q) \to (\neg p)$.

Si $\neg q$ est faux (ou q est vraie), l'implication est automatiquement vraie et il n'y a rien à montrer. (Cette phrase est souvent omise).

Supposons $\neg q$ est vraie, c.-à-d, q est fausse. Puis (avec cette hypothèse et avec de l'aide des théorèmes déjà montrés), on montre que p sera aussi fausse, ou que $\neg p$ est vraie.

[7] Voir [R, p. 164]

On aura montré que $P \to Q$ est vraie. $\square$

Remarque. Cette version est basée sur l'équivalence logique: $(p \to q) \Leftrightarrow (\neg q \to \neg p)$. Chacun constitue la structure d'une preuve valide. Avec une preuve directe on peut travailler avec l'hypothèse que P est vraie pour montrer qu'alors Q est aussi vraie. Mais de temps en temps il est plus facile de commencer avec l'hypothèse que Q est faux pour montrer qu'alors P est aussi faux.

Structure d'une preuve par l'absurde typique. Montrons $p \to q$ par une preuve par l'absurde. Supposons par contre que $p \to q$ est fausse, c.-à-d que p est vraie mais q est fausse.

Puis en utilisant ces deux hypothèses (que p est vraie et q fausse) (et les théorèmes déjà montrés) on montre qu'un certain proposition logique auxiliaire, disons r, est fausse (donc que $\neg r$ est vrai). Puis on remarque que c'était déjà connu (ou on montre directement sans utiliser l'hypothèse $p \wedge \neg q$) que r est vraie. Ce qui est absurde (ou une contradiction).

On conclut que $p \to q$ est vraie. $\square$

Remarque. C'est une variation sur une preuve par l'absurde, ou l'énoncé à montrer est $P := p \to q$ (à la place de p). On a $(p \to q) \Leftrightarrow (\neg p \vee q)$ et le contraire est $\neg(\neg p \vee q) \Leftrightarrow p \wedge \neg p$ (par De Morgan et double négation.

Différence entre la preuve directe, la preuve indirecte et la preuve par l'absurde pour montrer une implication $p \to q$? Pour la preuve directe: on peut utiliser l'hypothèse p pour montrer la conclusion q. Pour la preuve indirecte: on peut utiliser l'hypothèse $\neg q$ pour montrer la conclusion $\neg p$. Et pour une preuve par l'absurde: on peut utiliser deux hypothèses pour commencer le raisonnement (p vraie et q faux) pour dériver une absurdité (ou une contradiction). Ça dépend des propositions explicites p et q et de vos connaissance d'autres théorèmes quel façon est préférable!

Il y a aussi des fausse preuves, basées sur une contre-vérité. Considérons la "preuve":

Structure d'une fausse preuve typique. Il suffit de montrer $(\neg P) \to (\neg Q)$ est vraie.

Si $\neg P$ est faux (ou P est vraie), l'implication est automatiquement vraie. Donc il n'y a rien à faire.

Supposons $\neg P$ est vraie, ou P est fausse. Puis (avec cette hypothèse et avec de l'aide des théorèmes déjà montrés), on montre que Q sera aussi fausse, $\neg Q$ est vraie

On aura montré que $P \to Q$ est vraie. $\square$

Remarque. Pourquoi c'est une **fausse** preuve (donc pas une preuve du tout), car c'est basée sur une *contre-vérité*: les formules logiques $p \to q$ et $(\neg p) \to (\neq q)$ ne sont pas logiquement équivalentes.

Une telle FAUSSE preuve est inacceptable. (En fait, ce qu'on montrerait vraiment en fait est la *réciproque* $q \to p$).

4.14. **Si et seulement si.** Aussi utilisé souvent est l'équivalence logique entre "$(p \leftrightarrow q)$ et "$(p \to q) \wedge (q \to p)$".

Proof. Encore une fois montrée par un tableau de vérité :

p	q	$p \leftrightarrow q$	$q \to p$	$q \to p$	$((p \to q) \wedge (q \to p))$
V	V	V	V	V	V
V	F	F	F	V	F
F	V	F	V	F	F
F	F	V	V	V	V

On conclut la preuve, car les colonnes correspondantes sont identiques. $\square$

En mots: $P \leftrightarrow Q$ est vraie si et seulement si $P \to Q$ et sa réciproque $Q \to P$ sont vraies.

On conclut que pour montrer que $p \leftrightarrow q$ (est vraie) il suffit de montrer que $p \to q$ (est vraie) et puis de montrer qu'aussi $q \to p$ (est vraie) (ou alternativement que $(\neg p) \to (\neq q)$(est vraie)).

Département de mathématiques et de statistique, Université de Montréal, C.P. 6128, succursale Centre-ville, Montréal (Québec), Canada H3C 3J7

E-mail address: broera@DMS.UMontreal.CA

Logique

Exercice 1 :

Parmi les assertions suivantes, lesquelles sont vraies, lesquelles sont fausses et pourquoi ?

1. Si Napoléon était chinois alors $3 - 2 = 2$
2. Soit Cléopâtre était chinoise, soit les grenouilles aboient.
3. Soit les roses sont des animaux, soit les chiens ont 4 pattes.
4. Si l'homme est un quadrupède, alors il parle.
5. Les roses ne sont ni des animaux, ni des fleurs.
6. Paris est en France ou Madrid est en chine.
7. La pierre ponce est un homme si et seulement si les femmes sont des sardines.
8. Les poiriers ne donnent pas de melons, et Cléopâtre n'est pas chinoise.

Aller à : Correction exercice 1 :

Exercice 2 :

Soient (P), (Q) et (R) trois propositions, donner la négation de

a) (P) et $\big(\text{non}(Q) \text{ ou } (R)\big)$

b) $\big((P) \text{ et } (Q)\big) \Rightarrow (R)$

Aller à : Correction exercice 2 :

Exercice 3 :

Soient A, B et C trois assertions. Pour chacune des assertions suivantes :

$(A_1) \equiv \big(A \text{ et non}(B)\big)$; $(A_2) \equiv \big(A \text{ ou non}(B)\big)$; $(A_3) \equiv \big(A \text{ ou } (B \text{ et } C)\big)$; $(A_4) \equiv \big(A \text{ et } (B \text{ ou } C)\big)$

$(A_5) \equiv \big(A \Rightarrow \text{non}(B)\big)$; $(A_6) \equiv (A \Rightarrow B)$; $(A_7) \equiv (\text{non}(A \text{ ou } B) \Rightarrow C)$; $(A_8) \equiv \big((A \text{ et } B) \Rightarrow \text{non}(C)\big)$

Ecrire sa négation.

Aller à : Correction exercice 3 :

Exercice 4 :

Donner la négation mathématique des phrases suivantes

1. Toutes les boules contenues dans l'urne sont rouges.
2. Certains nombres entiers sont pairs.
3. Si un nombre entier est divisible par 4, alors il se termine par 4.

Soit $f : \mathbb{R} \to \mathbb{R}$

4. f est positive, c'est-à-dire « $\forall x \in \mathbb{R}, f(x) \geq 0$ »
5. f est paire sur $\mathbb{R}$, c'est-à-dire « $\forall x \in \mathbb{R}, f(-x) = f(x)$ »

Aller à : Correction exercice 4 :

Exercice 5 :

Soient les propositions, (P) « J'ai mon permis de conduire » et (Q) « j'ai plus de 18 ans »

Les propositions $(P) \Rightarrow (Q)$ et $(Q) \Rightarrow (P)$ sont-elles vraies ? Que peut-on conclure ?

Aller à : Correction exercice 5 :

Exercice 6 :

Parmi les assertions suivantes, lesquelles sont vraies, lesquelles sont fausses et pourquoi ?

1. $(2 < 3)$ et $(2 \text{ divise } 4)$
2. $(2 < 3)$ et $(2 \text{ divise } 5)$
3. $(2 < 3)$ ou $(2 \text{ divise } 5)$
4. $(2 < 3)$ et $\text{non}(2 \text{ divise } 5)$

5. $\text{non}(2 < 3)$ ou $(2 \text{ divise } 5)$

Aller à : Correction exercice 6 :

Exercice 7 :

Soient A et B deux parties de $\mathbb{N}$. Ecrire en utilisant $\forall, \exists$ les assertions

$$A = \emptyset, A \cap B \neq \emptyset, A \subset B, A \not\subset B$$

Aller à : Correction exercice 7 :

Exercice 8 :

On considère la proposition (P) suivante :

(P) « Pour tout nombre réel x, il existe au moins un entier naturel N supérieur ou égal à x »

1. Ecrire la proposition (P) avec des quantificateurs.
2. Ecrire la négation avec des quantificateurs puis l'énoncer en français.

Aller à : Correction exercice 8 :

Exercice 9 :

Notons E l'ensemble des étudiants, S l'ensemble des jours de la semaine et pour un étudiant x, $h_j(x)$ son heure de réveil le jour j.

a) Ecrire avec des symboles mathématiques la proposition « Tout étudiant se réveille au moins un jour de la semaine avant 8h »
b) Ecrire la négation de cette proposition avec des symboles mathématiques puis en français.

Aller à : Correction exercice 9 :

Exercice 10 :

Soit $\mathbb{P} \subset \mathbb{N}$ l'ensemble des nombres premiers et A une partie de $\mathbb{N}$. Ecrire en utilisant $\forall, \exists$ les assertions

A est une partie finie de $\mathbb{N}$, A est une partie infinie de $\mathbb{N}$.

Tout entier naturel $n \geq 2$ admet un diviseur premier, les éléments de A ont un diviseur premier commun, les éléments de A n'ont aucun diviseur premier commun.

Aller à : Correction exercice 10 :

Exercice 11 :

Soit n un entier naturel quelconque. Parmi les implications suivantes, lesquelles sont vraies, lesquelles sont fausses et pourquoi ? Donner leur contraposée et leur négation.

1. $\forall n \in \mathbb{N}, (n \geq 5) \Rightarrow (n > 3)$
2. $\forall n \in \mathbb{N}, (n \geq 5) \Rightarrow (n > 6)$
3. $\forall n \in \mathbb{N}, (n \geq 5) \Rightarrow (n \leq 6)$
4. $\forall n \in \mathbb{N}, (n < 1) \Rightarrow (2 \text{ divise } n)$
5. $\forall n \in \mathbb{N}, (n < 1) \Rightarrow (n \text{ divise } 2)$
6. $\forall n \in \mathbb{N}, (n < 2) \Rightarrow (n^2 = n)$

Aller à : Correction exercice 11 :

Exercice 12 :

Parmi les équivalences suivantes, lesquelles sont vraies, lesquelles sont fausses et pourquoi ?

1. $\forall n \in \mathbb{N}, (n \geq 5) \Leftrightarrow (n > 4)$
2. $\forall n \in \mathbb{N}, (n \geq 5) \Leftrightarrow (n \geq 4)$
3. $\forall n \in \mathbb{N}, \big((n \geq 5) \text{ et } (n \text{ divise } 12)\big) \Leftrightarrow (n = 6)$

Aller à : Correction exercice 12 :

Exercice 13 :

Soient les 4 assertions suivantes :

a. $\exists x \in \mathbb{R}\ \forall y \in \mathbb{R}\ x + y > 0$
b. $\forall x \in \mathbb{R}\ \exists y \in \mathbb{R}\ x + y > 0$
c. $\forall x \in \mathbb{R}\ \forall y \in \mathbb{R}\ x + y > 0$
d. $\exists x \in \mathbb{R}\ \forall y \in \mathbb{R}\ y^2 > x$

1. Les assertions a, b, c et d sont-elles vraies ou fausses ?
2. Donner leur négation

Aller à : Correction exercice 13 :

Exercice 14 :

Soit $(q_n)_{n\in\mathbb{N}}$ une suite de nombres rationnels. Que signifie en mots les assertions suivantes

$\forall n \in \mathbb{N}, \exists l \in \mathbb{Z}, q_n = l,$

$$\exists l \in \mathbb{Z}, \forall n \in \mathbb{N}, q_n = l, \quad \forall l \in \mathbb{Z}, \exists n \in \mathbb{N}, q_n = l, \quad \forall q \in \mathbb{Q}_{>0}, \forall n \in \mathbb{N}, |q_n| < q$$

Attention : il ne s'agit pas de faire la lecture à voix haute de ces quatre suites de symboles mais de traduire l'énoncé en phrase courte dont la compréhension est immédiate.

Aller à : Correction exercice 14 :

Exercice 15 :

1. Donner la négation de la phrase mathématique suivante :

$$\forall \epsilon > 0, \exists N \in \mathbb{N}, \forall n \in \mathbb{N}, \forall p \in \mathbb{N}, n \geq N \text{ et } p \geq 0 \Rightarrow |u_{n+p} - u_n| < \epsilon$$

2. Donner la contraposée de la phrase mathématique suivante :

$$\forall \epsilon > 0, \exists N \in \mathbb{N}, \forall n \in \mathbb{N}, \forall p \in \mathbb{N}, n \geq N \text{ et } p \geq 0 \Rightarrow |u_{n+p} - u_n| < \epsilon$$

Aller à : Correction exercice 15 :

Exercice 16 :

Soient $x_0 \in \mathbb{R}$ et f une application de $\mathbb{R}$ dans $\mathbb{R}$.

$$\forall \epsilon > 0, \exists \alpha > 0, \forall x \in \mathbb{R}, \qquad |x - x_0| < \alpha \Rightarrow |f(x) - f(x_0)| < \epsilon$$

Donner la négation et la contraposée de cette phrase logique.

Allez à : Correction exercice 16 :

Exercice 17 :

Compléter, lorsque c'est possible, avec $\forall$ ou $\exists$ pour que les énoncés suivants soient vrais.

a) $\ldots x \in \mathbb{R}, (x+1)^2 = x^2 + 2x + 1$
b) $\ldots x \in \mathbb{R}, x^2 + 3x + 2 = 0$
c) $\ldots x \in \mathbb{R}, 2x + 1 = 0$
d) $\ldots x \in \mathbb{R}, x^2 + 2x + 3 \neq 0$

Aller à : Correction exercice 17 :

Exercice 18 :

Les propositions suivantes sont-elles vraies ? Lorsqu'elles sont fausses, énoncer leur négation.

a) $\exists x \in \mathbb{N}, x^2 > 7$
b) $\forall x \in \mathbb{N}, x^2 > 7$
c) $\forall x \in \mathbb{N}, \exists y \in \mathbb{N}, y > x^2$
d) $\exists x \in \mathbb{N}, \forall\, y \in \mathbb{N}, y > x^2$

Aller à : Correction exercice 18 :

Corrections

Correction exercice 1 :

1. Il s'agit, ici d'une implication. « Napoléon est chinois » est faux et « $3-2=2$ » est faux, or la seule possibilité pour qu'une implication soit fausse est qu'une assertion vraie implique une assertion fausse, donc l'assertion 1. est vraie.
2. Une phrase, en français, du genre « soit …, soit … » se traduit mathématiquement par « … ou … » « Cléopâtre était chinoise » est faux et « les grenouilles aboient » est faux donc l'assertion 2. est fausse.
3. « les roses sont des animaux » est faux et « les chiens ont 4 pattes » est vrai, donc l'assertion 3. est vraie.
4. « l'homme est un quadrupède » est faux et «il parle » est vrai, donc l'assertion 4. est vraie.
5. « les roses ne sont ni des animaux, ni des fleurs » peut se traduire par « les roses ne sont pas des animaux et les roses ne sont pas des fleurs ». « les roses ne sont pas des animaux » est vrai et « les roses ne sont pas des fleurs » est faux donc « les roses ne sont ni des animaux, ni des fleurs » est faux. Avec un minimum de bon sens c'est assez évident !
6. « Paris est en France » est vrai et « Madrid est en chine » est faux, donc « Paris est en France ou Madrid est en chine » est vrai.
7. « la pierre ponce est un homme » est faux et «les femmes sont des sardines » est faux, une équivalence entre deux assertion fausse est vraie.
8. « les poiriers ne donnent pas de melons » est vrai et «Cléopâtre n'est pas chinoise » est vrai, donc « les poiriers ne donnent pas de melons, et Cléopâtre n'est pas chinoise » est vrai.

Aller à : Exercice 1 :

Correction exercice 2 :

a)

$$\begin{aligned}\text{non}\big((P)\ \text{et}\ (\text{non}(Q)\ \text{ou}\ (R))\big) &\equiv \big(\text{non}(P)\ \text{ou}\ \text{non}(\text{non}(Q)\text{ou}\ (R))\big)\\ &\equiv (\text{non}(P)\ \text{ou}\ ((Q)\ \text{et}\ \text{non}(R))\\ &\equiv \big(\text{non}(P)\ \text{ou}\ (Q)\big)\ \text{et}\ \big(\text{non}(P)\ \text{ou}\ \text{non}(R)\big)\\ &\equiv \big(\text{non}(P)\ \text{ou}\ (Q)\big)\ \text{et}\ \text{non}\big((\text{P})\ \text{et}\ (R)\big)\end{aligned}$$

Les deux dernières équivalences logiques me paraissent acceptables, parce qu'il y a souvent différentes façon d'exprimer une négation, ensuite il faut voir dans les exercices comment se présentent les propositions (P), (Q) et (R).

b)

$$\text{non}\big(((P)\ \text{et}\ (Q)) \Rightarrow (R)\big) \equiv \big((P)\ \text{et}\ (Q)\big)\ \text{et}\ \text{non}(\text{R}) \equiv (P)\ \text{et}\ (Q)\ \text{et}\ \text{non}(\text{R})$$

Aller à : Exercice 2 :

Correction exercice 3 :

$$\text{non}(A_1) \equiv \text{non}\big(A\ \text{et}\ \text{non}(B)\big) \equiv \text{non}(A)\ \text{ou}\ \text{non}\big(\text{non}(B)\big) \equiv \text{non}(A)\ \text{ou}\ B$$

$$\text{non}(A_2) \equiv \text{non}\big(A\ \text{ou}\ \text{non}(B)\big) \equiv \text{non}(A)\ \text{et}\ \text{non}\big(\text{non}(B)\big) \equiv \text{non}(A)\ \text{et}\ B$$

$$\begin{aligned}\text{non}(A_3) &\equiv \text{non}\big(A\ \text{ou}\ (B\ \text{et}\ C)\big) \equiv \text{non}(A)\ \text{et}\ \text{non}(B\ \text{et}\ C) \equiv \text{non}(A)\ \text{et}\ \big(\text{non}(B)\ \text{ou}\ \text{non}(C)\big)\\ &\equiv \big(\text{non}(A)\ \text{et}\ \text{non}(B)\big)\ \text{ou}\ \big(\text{non}(A)\ \text{et}\ \text{non}(C)\big)\end{aligned}$$

Il y a d'autres expressions possibles de cette négation.

$$\begin{aligned}\text{non}(A_4) &\equiv \text{non}\big(A\ \text{et}\ (B\ \text{ou}\ C)\big) \equiv \text{non}(A)\ \text{ou}\ \text{non}(B\ \text{ou}\ C) \equiv \text{non}(A)\ \text{ou}\ \big(\text{non}(B)\ \text{et}\ \text{non}(C)\big)\\ &\equiv \big(\text{non}(A)\ \text{ou}\ \text{non}(B)\big)\ \text{et}\ \big(\text{non}(A)\ \text{ou}\ \text{non}(C)\big)\end{aligned}$$

Il y a d'autres expressions possibles de cette négation.

$$\text{non}(A_5) \equiv \text{non}\big(A \Rightarrow \text{non}(B)\big) \equiv \text{non}\big(\text{non}(A) \text{ ou } \text{non}(B)\big) \equiv \text{A et } B$$
$$\text{non}(A_6) \equiv \text{non}(A \Rightarrow B) \equiv \text{non}\big(\text{non}(A) \text{ ou } (\text{B})\big) \equiv (A) \text{ et non}(B)$$
$$\text{non}(A_7) \equiv \text{non}\big((\text{non}(A \text{ ou } B) \Rightarrow C)\big) \equiv \text{non}\big(\text{non}\big(\text{non}(A \text{ ou } B)\big) \text{ ou } C\big) \equiv \text{non}\big((A \text{ ou } B) \text{ ou } C\big)$$
$$\equiv \text{non}(A \text{ ou } B) \text{ et non}(C) \equiv \text{non}(A) \text{ et non}(B) \text{ et non}(C)$$
$$\text{non}(A_8) = \text{non}\big((A \text{ et } B) \Rightarrow \text{non}(C)\big) \equiv \text{non}\Big(\text{non}(A \text{ et } B) \text{ et non}\big(\text{non}(C)\big)\Big)$$
$$\equiv \text{non}(\text{non}(A \text{ et } B) \text{ et C}) \equiv (A \text{ et } B) \text{ ou non}(C) = \big(A \text{ ou non}(C)\big) \text{ et } (B \text{ ou non}(C)$$

Aller à : Exercice 3 :

Correction exercice 4 :

1. Il existe une boule qui n'est pas rouge dans l'urne. (La négation de « pour tout » est « il existe » et la négation « rouge » est « n'est pas rouge »).
2. Tous les nombres entiers sont pairs. (La négation de « il existe » (dans l'énoncé « certains » signifie « il existe ») est « tous ». Dans cette question on ne se demande pas si la proposition est vraie ou fausse.
3. Il s'agit d'une implication, la négation de $(P) \Rightarrow (Q)$ est : (P) et $\text{non}(Q)$ donc la négation demandée est « un nombre entier est divisible par 4 et il ne se termine pas par 4 ». Dans cette question on ne se demande pas si l'implication est vraie ou fausse.
4. $\exists x \in \mathbb{R}$ tel que $f(x) < 0$.
5. $\exists x \in \mathbb{R}, f(-x) \neq f(x)$

Aller à : Exercice 4 :

Correction exercice 5 :

$(P) \Rightarrow (Q)$ est vraie, par contre $(Q) \Rightarrow (P)$ est fausse, on en conclut que ces deux propositions ne sont pas équivalentes.

Aller à : Exercice 5 :

Correction exercice 6 :

1. $(2 < 3)$ est vrai et $(2 \text{ divise } 4)$ est vrai donc $(2 < 3)$ et $(2 \text{ divise } 4)$ est vrai.
2. $(2 < 3)$ est vrai et $(2 \text{ divise } 5)$ est faux, l'un des deux est faux donc $(2 < 3)$ et $(2 \text{ divise } 5)$ est faux.
3. $(2 < 3)$ est vrai et $(2 \text{ divise } 5)$ est faux, l'un des deux est vrai donc $(2 < 3)$ et $(2 \text{ divise } 5)$ est vrai.
4. $(2 < 3)$ est vrai et $\text{non}(2 \text{ divise } 5)$ est vrai, les deux sont vrais donc $(2 < 3)$ et $\text{non}(2 \text{ divise } 5)$ est vrai.
5. $(2 < 3)$ est vrai donc $\text{non}(2 < 3)$ est faux (on peut aussi dire que $\text{non}(2 < 3) \Leftrightarrow (2 \geq 3)$ qui est faux) et $(2 \text{ divise } 5)$ est faux par conséquent $\text{non}(2 < 3)$ ou $(2 \text{ divise } 5)$ est faux car les deux assertions sont fausses.

Aller à : Exercice 6 :

Correction exercice 7 :

$$\forall x \in \mathbb{N}, x \notin A, \qquad \exists x \in \mathbb{N}, x \in A, x \in B, \quad \forall x \in A, x \in B, \quad \exists x \in A, x \notin B$$

Aller à : Exercice 7 :

Correction exercice 8 :

1. $\forall x \in \mathbb{R}, \exists N \in \mathbb{N}, N \geq x$
2. $\exists x \in \mathbb{R}, \forall N \in \mathbb{N}, N < x$. Il existe un réel tel que pour tout N entier, N est strictement inférieur à x.

Aller à : Exercice 8 :

Correction exercice 9 :

a) $\forall x \in E, \exists j \in S, h_j(x) < 8h$.

b) $\exists x \in E, \forall j \in S, h_j(x) \geq 8h$. Il y a un étudiant qui se lève à 8h ou après 8h tous les jours de la semaine. (Donc c'est un gros fainéant).

Aller à : Exercice 9 :

Correction exercice 10 :

$$\exists M \in \mathbb{N}, \forall n \in A, n < M, \qquad \forall M \in \mathbb{N}, \exists n \in A, n \geq M$$

$$\forall n \in \mathbb{N}, \exists p \in \mathbb{P}, \exists k \in \mathbb{N}, n = kp, \qquad \exists p \in \mathbb{P}, \forall n \in A, \exists k \in \mathbb{N}, n = kp, \qquad \forall p \in \mathbb{P}, \exists n \in A, \forall k \in \mathbb{N}, n \neq kp$$

Aller à : Exercice 10 :

Correction exercice 11 :

1. $\forall n \in \mathbb{N}, (n \geq 5) \Rightarrow (n > 3)$ est vraie.
   Sa contraposée est $\forall n \in \mathbb{N}, (n \leq 3) \Rightarrow (n < 5)$ ou encore $\forall n \in \mathbb{N}, (n < 4) \Rightarrow (n \leq 4)$.
   (On rappelle que $\big((P) \Rightarrow (Q)\big) \equiv \big(\text{non}(P) \text{ ou } (Q)\big)$ donc la négation de $\big((P) \Rightarrow (Q)\big)$ est
   $$\text{non}\Big(\big(\text{non}(P) \text{ ou } (Q)\big)\Big) \equiv \Big(\text{non}\big(\text{non}(P)\big) \text{ et } \text{non}(Q)\Big) \equiv \big((P) \text{ et } \text{non}(Q)\big)$$
   $$\text{non}\big((n \geq 5) \Rightarrow (n > 3)\big) \equiv \big((n \geq 5) \text{ et } (n \leq 3)\big) \equiv \big((n > 4) \text{ et } (n < 4)\big)$$
   La négation est : $\exists n \in \mathbb{N}, (n \geq 5) \text{ et } (n \leq 3)$
2. $\forall n \in \mathbb{N}, (n \geq 5) \Rightarrow (n > 6)$ est faux car pour $n = 5$, $(n \geq 5)$ est vrai et $(n > 6)$ est faux (idem pour $n = 6$).
   Sa contraposée est $\forall n \in \mathbb{N}, (n \leq 6) \Rightarrow (n < 5) \equiv (n < 7) \Rightarrow (n < 5)$.
   Sa négation est $\big(\exists n \in \mathbb{N}, (n \geq 5) \text{ et } (n \leq 6)\big) \equiv \big(\exists n \in \mathbb{N}, (n > 4) \text{ et } (n < 7)\big)$
3. $\forall n \in \mathbb{N}, (n \geq 5) \Rightarrow (n \leq 6)$ est faux car pour $n = 7$, $(n \geq 5)$ est vrai et $(n \leq 6)$ est faux.
   Sa contraposée est $\forall n \in \mathbb{N}, (n \leq 6) \Rightarrow (n < 5) \equiv (n < 7) \Rightarrow (n < 5)$.
   Sa négation est
   $$\big(\exists n \in \mathbb{N}, (n \geq 5) \text{ et } (n > 6)\big) \equiv \big(\exists n \in \mathbb{N}, (n > 4) \text{ et } (n > 7)\big) \equiv \exists n \in \mathbb{N}, (n > 7)$$
4. $\forall n \in \mathbb{N}, (n < 1) \equiv (n = 0)$ si $(n < 1)$ est vrai alors $n = 0$ et comme $0 = 0 \times 2$, cela signifie que 2 divise 0, par conséquent $\forall n \in \mathbb{N}, (n < 1) \Rightarrow \forall n \in \mathbb{N}, (n \text{ divise } 2)$ est vrai.
   Il n'y a que cela à vérifier parce que si $n < 1$ est faux, quoiqu'il arrive à la conclusion, l'implication est vraie.
   On aura pu aussi voir que :
   $$(n \text{ divise } 2) \equiv (\exists k \in \mathbb{N}, n = 2k) \equiv (n \text{ est pair})$$
   Sa contraposée est $\big(\forall n \in \mathbb{N}, (n \text{ ne divise pas } 2) \Rightarrow (n \geq 1)\big) \equiv \forall n \in \mathbb{N}, (n \text{ est impair}) \Rightarrow (n \geq 1)$.
   Vu ainsi il est clair que la contraposée est vraie et que donc $\forall n \in \mathbb{N}, (n < 1) \Rightarrow (n \text{ divise } 2)$ est vrai.
   La négation est : $\exists\, n \in \mathbb{N}, (n < 1)$ et $(n \text{ ne divise pas } 2)$
5. Comme dans le 4. $\forall n \in \mathbb{N}, (n < 1) \equiv (n = 0)$ mais 0 ne divise pas 2, sinon cela signifierait qu'il existe $k \in \mathbb{N}$ tel que $2 = k \times 0$ ce qui est faux par conséquent $\forall n \in \mathbb{N}, (n < 1) \Rightarrow (n \text{ divise } 2)$ est faux. En effet une assertion vraie ne peut pas impliquer une assertion fausse.
   La contraposée est $\forall n \in \mathbb{N}, (n \text{ ne divise pas } 2) \Rightarrow (n \geq 1)$.
   La négation est $\big(\exists n \in \mathbb{N}, (n < 1) \text{ et } (n \text{ ne divise pas } 2)\big)$. Vérifions que cette implication est vraie : soit $n = 0$ et $(n < 1)$ est vrai et $(0 \text{ ne divise pas } 2)$ est vrai ce qui entraine que $\big(\exists n \in \mathbb{N}, (n < 1) \text{ et } (n \text{ ne divise pas } 2)\big)$ est vrai.
6. $\forall n \in \mathbb{N}, (n < 2) \equiv (n \in \{0,1\})$, si $n = 0$ alors $n^2 = 0^2 = 0 = n$ et si $n = 1$ alors $n^2 = 1^2 = 1 = n$ donc $\forall n \in \mathbb{N}, (n < 2) \Rightarrow (n^2 = n)$.
   La contraposée est $\forall n \in \mathbb{N}, (n^2 \neq n) \Rightarrow (n \geq 2)$

Sa négation est $\exists n \in \mathbb{N}, (n < 2)$ et $(n^2 \neq n)$.

Aller à : Exercice 11 :

Correction exercice 12 :

1. $\forall n \in \mathbb{N}, (n \geq 5) \Leftrightarrow (n > 4)$ car un entier strictement supérieur à 4 est supérieur ou égal à 5.
2. $\forall n \in \mathbb{N}, (n \geq 5) \Leftrightarrow (n \geq 4)$ est faux car pour $n = 4$, $(n \geq 4)$ est vrai et $(n \geq 5)$ est faux.
3. Les diviseurs entiers et positifs de 12 sont $\{1,2,3,4,6,12\}$ donc les diviseurs entiers et supérieurs ou égaux à 5 sont 6 et 12, bref, il suffit de dire que 12 rend vrai $\big((n \geq 5)$ et $(n$ divise $12)\big)$ et faux $(n = 6)$ pour pouvoir affirmer que $\forall n \in \mathbb{N}, \big((n \geq 5)$ et $(n$ divise $12)\big) \Leftrightarrow (n = 6)$ est faux.

Aller à : Exercice 12 :

Correction exercice 13 :

1. a. est faux car si un tel x existe, il suffit de prendre $y = -x - 1$ pour que $x + y > 0$ soit faux, en effet $x + (-x - 1) = -1 < 0$
   b. est vrai, car pour un x fixé, on choisit $y = -x + 1$ de façon à ce que $x + (-x + 1) = 1 > 0$.
   c. est faux car si on prend $x = y = -1$ alors $x + y = -2$ est faux et donc on n'a pas $x + y > 0$
   d. Il suffit de prendre $x = -1$, ainsi pour tout $y \in \mathbb{R}$, $y^2 > -1$, l'assertion est vraie.
2. a. $\forall x \in \mathbb{R}\ \exists\ y \in \mathbb{R}\ x + y \leq 0$
   (on pourra montrer, à titre d'exercice que cette assertion quantifiée est vraie).
   b. $\exists x \in \mathbb{R}\ \forall y \in \mathbb{R}\ x + y \leq 0$
   (on pourra montrer, à titre d'exercice que cette assertion quantifiée est fausse).
   c. $\exists x \in \mathbb{R}\ \exists y \in \mathbb{R}\ x + y \leq 0$
   (on pourra montrer, à titre d'exercice que cette assertion quantifiée est vraie).
   d. $\forall x \in \mathbb{R}\ \exists y \in \mathbb{R}\ y^2 \leq x$

Aller à : Exercice 13 :

Correction exercice 14 :

La suite $(q_n)_{n\in\mathbb{N}}$ est une suite d'entiers.
La suite $(q_n)_{n\in\mathbb{N}}$ est constante dont la valeur est entière.
La suite $(q_n)_{n\in\mathbb{N}}$ prend toutes les valeurs entières.
La suite $(q_n)_{n\in\mathbb{N}}$ est constante et vaut 0.

Aller à : Exercice 14 :

Correction exercice 15 :

1.
$$\exists \epsilon > 0, \forall N \in \mathbb{N}, \exists n \in \mathbb{N}, \exists p \in \mathbb{N}, n \geq N \text{ et } p \geq 0 \text{ et } \left|u_{n+p} - u_n\right| \geq \epsilon$$
2.
$$\forall \epsilon > 0, \exists N \in \mathbb{N}, \forall n \in \mathbb{N}, \forall p \in \mathbb{N}, \left|u_{n+p} - u_n\right| \geq \epsilon \Rightarrow n < N \text{ ou } p < 0$$

Aller à : Exercice 15 :

Correction exercice 16 :

La négation est :
$$\exists \epsilon > 0, \forall \alpha > 0, \exists x \in \mathbb{R}, \quad |x - x_0| < \alpha \quad \text{et} \quad |f(x) - f(x_0)| \geq \epsilon$$
La contraposée est
$$\forall \epsilon > 0, \exists \alpha > 0, \forall x \in \mathbb{R}, |x - x_0| < \alpha, \qquad |f(x) - f(x_0)| \geq \epsilon \Rightarrow |x - x_0| \geq \alpha$$

Allez à : Exercice 16 :

Correction exercice 17 :

a) $\forall x \in \mathbb{R}, (x+1)^2 = x^2 + 2x + 1$
b) $\exists x \in \mathbb{R}, x^2 + 3x + 2 = 0$
c) $\exists x \in \mathbb{R}, 2x + 1 = 0$
d) $\exists x \in \mathbb{R}, x^2 + 2x + 3 \neq 0$ ou $\forall x \in \mathbb{R}, x^2 + 2x + 3 \neq 0$

Aller à : Exercice 17 :

Correction exercice 18 :

a) Vraie
b) Fausse par exemple pour $x = 1$, la négation est : $\exists x \in \mathbb{N}, x^2 \leq 7$
c) Vraie
d) Fausse car la négation est manifestement vraie, la négation est : $\forall x \in \mathbb{N}, \exists y \in \mathbb{N}, y \geq x^2$.

Aller à : Exercice 18 :

Introduction à la logique
Correction des exercices

Auteur : Antoine Zimmermann
Tronc commun informatique – Mines Saint-Étienne

Ces corrections ont été rédigées par Antoine Zimmermann sur des suggestions de réponses de Gauthier Picard (anciennement Mines Saint-Étienne).

Table des matières

1. Logique des propositions
    1. Exercice 1.6.1
    2. Exercice 1.6.2
    3. Exercice 1.6.3
    4. Exercice 1.6.4
    5. Exercice 1.6.5
    6. Exercice 1.6.6
    7. Exercice 1.6.7
    8. Exercice 1.6.8
2. Logique des prédicats (du premier ordre)
    1. Exercice 2.5.1
    2. Exercice 2.5.2
    3. Exercice 2.5.3
    4. Exercice 2.5.4
    5. Exercice 2.5.5
    6. Exercice 2.5.6
    7. Exercice 2.5.7
3. Principe de résolution
    1. Exercice 3.5.1
    2. Exercice 3.5.2
    3. Exercice 3.5.3
    4. Exercice 3.5.4
    5. Exercice 3.5.5
    6. Exercice 3.5.6

Chapitre 1

Logique des propositions

Exercice 1.6.1 — En utilisant les atomes S (le soleil brille), P (il pleut), B (il bruine), A (il y a un arc en ciel), O (il a un vent d'Ouest), E (il y a du vent d'Est), traduire dans la logique des propositions $\mathscr{L}_0$ les énoncés suivants :

(a) S'il pleut et que le soleil brille en même temps, alors il y a un arc en ciel,
(b) Si le vent d'ouest amène la pluie, on n'a jamais vu qu'un vent d'est soit porteur de pluie,
(c) La bruine est une forme de pluie.

Les formules suivantes peuvent modéliser les 3 phrases de l'énoncé. La phrase (b) est difficile à modéliser complètement car elle cache une relation de causalité. Si le vent d'est n'apporte pas la pluie (le vent d'est n'est pas la cause de la pluie), cela ne veut pas forcément dire qu'il ne pleut pas lorsqu'il y a un vent d'est. Mais s'il peut pleuvoir lorsqu'il y a un vent d'est, alors on ne peut rien dire sur le sujet en logique propositionnelle. En revanche, il serait faux d'écrire $((O \to P) \wedge \neg(E \to P))$ car cela implique logiquement qu'il ne pleut pas, nécessairement. Autrement dit, cette dernière modélisation décrit un monde où il ne peut pas pleuvoir.

(a) $((S \wedge P) \to A)$
(b) $((O \to P) \wedge (E \to \neg P))$
(c) $(B \to P)$

Exercice 1.6.2 — Représenter dans le formalisme de la logique des propositions les théorèmes de géométrie suivants :

(a) Si un triangle est équilatéral alors il est isocèle.
(b) Un triangle rectangle n'est jamais équilatéral.
(c) Un carré est à la fois un parallélogramme et un rectangle.
(d) Un losange n'est ni un quadrilatère rectangle ni un triangle.
(e) Deux droites coplanaires sont soit sécantes, soit parallèles.
(f) Deux droites ne peuvent être à la fois sécantes et parallèles.

Pour cet exercice, il faut définir des propositions. Si on les définit de façon naïve, on peut aboutir à des aberrations. « être isocèle » n'est pas une affirmation. « Un triangle est isocèle » est une affirmation mais dépend de quel triangle on parle (ou bien cela signifie-t-il « l'un des triangles est isocèle » ? ou encore « il existe un triangle isocèle » ?). On considèrera donc une forme géométrique donnée et on considèrera les proposition : *tequi* pour « la forme est un triangle équilatéral » ; *tiso* pour « la forme est un triangle isocèle » ; *trec* pour « la forme est un triangle rectangle » ; *carr* pour « la forme est un carré » ; *pgram* pour « la forme est un parallélogramme » ; *rect* pour « la forme est un rectangle » ; *los* pour « la forme est un losange » ; *qrec* pour « la forme est un quadrilatère rectangle » ; *tri* pour « la forme est un triangle » ; *cop* pour « la forme est deux droites coplanaires » ; *sec* pour « la forme est deux droites séquantes » ; *para* pour « la forme est deux droites parallèles ».

(a) $(tequi \to tiso)$
(b) $(trec \to \neg tequi)$
(c) $(carr \to (pgram \wedge rect))$
(d) $(los \to (\neg qrec \wedge \neg tri))$
(e) $(cop \to ((sec \wedge \neg para) \vee (\neg sec \wedge para)))$
(f) $\neg(sec \wedge para)$

Dans le cas où l'on voudrait s'intéresser aux propriétés de plusieurs formes géométriques, par exemple d'un millier de triangles, pour savoir lesquelles sont de tel ou tel type, il faudrait introduire autant de formules propositionnelles qu'il y a de formes. Par exemple, pour des triangles t_1, …, t_{1000}, il faudrait écrire $(tequi_{t_1} \to tiso_{t_1})$, …, $(tequi_{t_{1000}} \to tiso_{t_{1000}})$. Chaque nouvelle forme introduite dans le système devrait être accompagnée d'une telle formule. À la place, on aimerait pouvoir écrire que « *pour tout triangle t,* $(tequi_t \to tiso_t)$ », mais pour être totalement général, il faudrait une infinité de formules propositionnelles. C'est pourquoi la logique propositionnelle n'est pas suffisante et qu'on introduit la logique du premier ordre dans le chapitre suivant.

Exercice 1.6.3 — Formaliser dans $\mathscr{L}_0$ les propositions suivantes :

(a) Si Alice et Julie viennent à Paris, Zoé viendra aussi ;
(b) Si Julie vient à Paris, Alice aussi ;
(c) Julie ou Zoé, l'une des deux au moins, viendra à Paris.

Alice viendra-t-elle à Paris ? Et Julie ? Et Zoé ?

Soient les variables propositionnelles :

- a : « Alice vient à Paris »
- j : « Julie vient à Paris »
- z : « Zoé vient à Paris »

« Si Alice et Julie viennent à Paris, Zoé viendra aussi » se traduit par $a \wedge j \to z$. « Si Julie vient à Paris, Alice aussi » se traduit par $j \to a$. « Julie ou Zoé, l'une des deux au moins, viendra à Paris » se traduit par $j \vee z$.

Les données se traduisent par la formule $F = (a \wedge j \to z) \wedge (j \to a) \wedge (j \vee z)$.

Pour savoir si Alice, Julie et Zoé viennent à Paris, nous utilisons la méthode des tables de vérité.

a	j	z	$(a \wedge j)$	$(a \wedge j \to z)$	$(j \to a)$	$(j \vee z)$	F
$\bot$	$\bot$	$\bot$	$\bot$	$\top$	$\top$	$\bot$	$\bot$
$\bot$	$\bot$	$\top$	$\bot$	$\top$	$\top$	$\top$	$\top$
$\bot$	$\top$	$\bot$	$\bot$	$\top$	$\bot$	$\top$	$\bot$
$\bot$	$\top$	$\top$	$\bot$	$\top$	$\bot$	$\top$	$\bot$
$\top$	$\bot$	$\bot$	$\bot$	$\top$	$\top$	$\bot$	$\bot$
$\top$	$\bot$	$\top$	$\bot$	$\top$	$\top$	$\top$	$\top$
$\top$	$\top$	$\bot$	$\top$	$\bot$	$\top$	$\top$	$\bot$
$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$

On peut en conclure que :

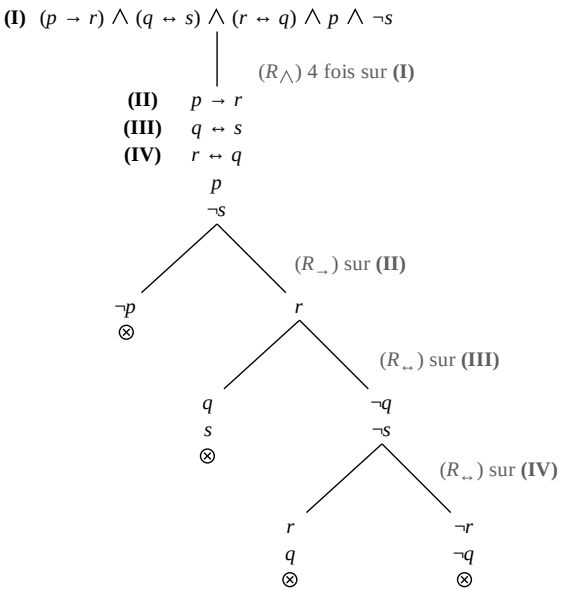
- Il se peut qu'Alice vienne à Paris ;
- Il se peut que Julie vienne à Paris ;
- Il est certain que Zoé viendra à Paris.

Exercice 1.6.4 — Montrer que $((p \to r) \wedge (q \leftrightarrow s) \wedge (r \leftrightarrow q) \wedge p) \vDash s$.

On utilise ici la méthode des tableaux. Cela revient à montrer que l'arbre de racine :

$$(p \to r) \wedge (q \leftrightarrow s) \wedge (r \leftrightarrow q) \wedge p \wedge \neg s$$

est fermé.



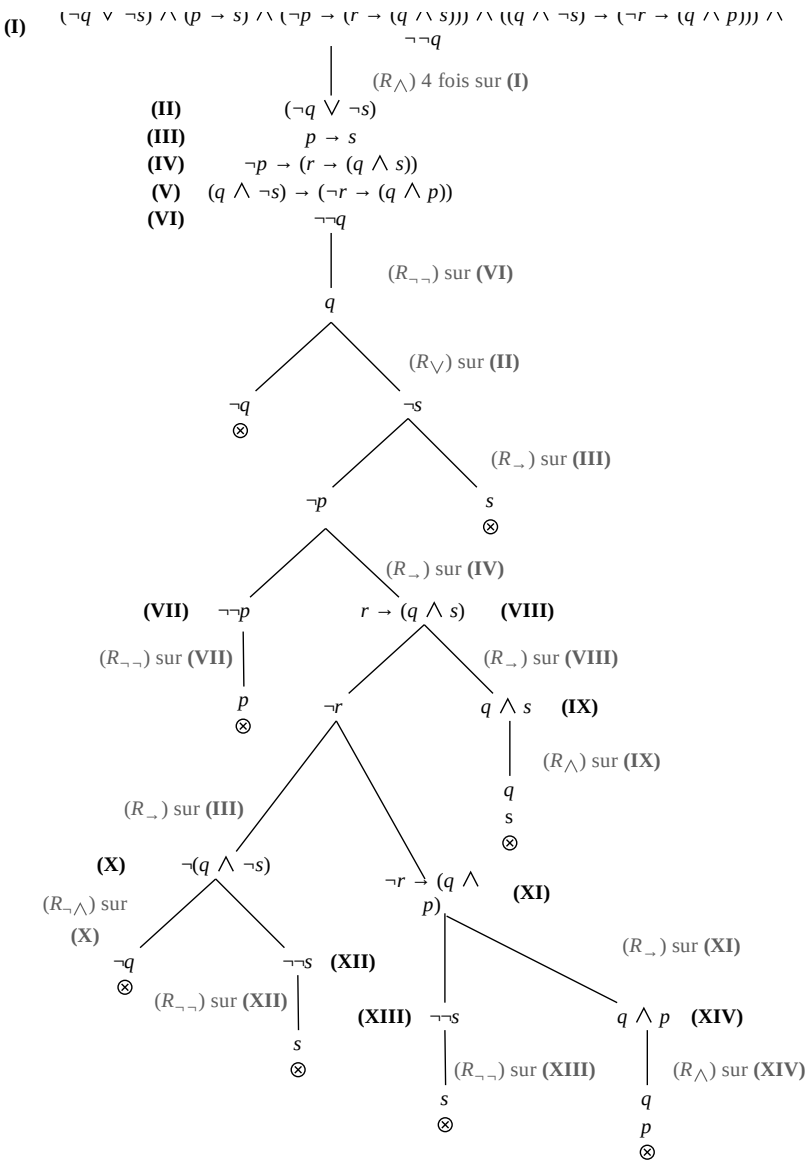
Exercice 1.6.5 — Montrer que :

$$((\neg q \vee \neg s) \wedge (p \to s) \wedge (\neg p \to (r \to (q \wedge s))) \wedge ((q \wedge \neg s) \to (\neg r \to (q \wedge p)))) \vDash \neg q.$$

On utilise ici la méthode des tableaux. Cela revient à montrer que l'arbre de racine :

$$\neg(\neg q \vee \neg s) \wedge (p \to s) \wedge (\neg p \to (r \to (q \wedge s))) \wedge ((q \wedge \neg s) \to (\neg r \to (q \wedge p))) \wedge \neg\neg q$$

est fermé.



Exercice 1.6.6 — Construire dans SF_0 la preuve de $\vdash ((a \to b) \to ((b \to c) \to (a \to c)))$.

D'après le théorème de la déduction, avoir une preuve de $\vdash ((a \to b) \to ((b \to c) \to (a \to c)))$ revient à prouver que $(a \to b) \vdash ((b \to c) \to (a \to c))$. En appliquant à nouveau le théorème de la déduction, cela revient à trouver une preuve de $(a \to b), (b \to c) \vdash (a \to c)$. En faisant une dernière fois appel au théorème de la déduction, cela revient à trouver une preuve de $(a \to b)$, $(b \to c)$, $a \vdash c$.

On obtient une preuve de c en posant :

$c_1 = a$	Hypothèse
$c_2 = (a \to b)$	Hypothèse
$c_3 = (b \to c)$	Hypothèse
$c_4 = b$	Modus ponens sur c_1 et c_2
$c_5 = c$	Modus ponens sur c_3 et c_4

Remarque — À proprement parler, nous ne construisons pas une preuve de $\vdash ((a \to b) \to ((b \to c) \to (a \to c)))$ mais nous démontrons l'existence d'une telle preuve par l'équivalence énoncée par le théorème de la déduction. Si nous avions accès à une preuve constructive du théorème de la déduction, nous pourrions partir de la preuve ci-dessus pour construire la preuve du théorème considéré.

Exercice 1.6.7 — Construire dans SF_0 la preuve de $\vdash (\neg a \to (a \to b))$.

D'après le théorème de la déduction, avoir une preuve de $\vdash (\neg a \to (a \to b))$ revient à prouver que $\neg a \vdash (a \to b)$.

On obtient une preuve de $(a \to b)$ en posant :

c_1	$= \neg a$	Hypothèse
c_2	$= (\neg a \to (\neg b \to \neg a))$	Axiome A_1 avec $A = \neg a$ et $B = \neg b$
c_3	$= (\neg b \to \neg a)$	Modus ponens sur c_1 et c_2
c_4	$= ((\neg b \to \neg a) \to (a \to b))$	Axiome A_3 avec $A = b$ et $B = a$
c_5	$= (a \to b)$	Modus ponens sur c_3 et c_4

Remarque — La même remarque que pour l'exercice 1.6.6 vaut pour cet exercice.

Exercice 1.6.8 — Lors de ses aventures au pays des merveilles rapportées par Lewis Carroll, Alice est souvent accompagnée par le chat de Cheshire. Ce félin énigmatique s'exprime sous la forme d'affirmations logiques qui sont toujours vraies. Alice se trouve dans un corridor dont toutes les portes à sa taille sont fermées. La seule porte ouverte est nettement trop petite pour qu'elle puisse l'emprunter. Une étagère est fixée au-dessus de cette porte. Le chat dit alors à Alice : « *L'un des flacons posés sur cette étagère contient un liquide qui te permettra de prendre une taille plus adéquate. Mais attention, les autres flacons peuvent contenir un poison fatal.* » Trois flacons sont effectivement posés sur l'étagère. Le premier est rouge, le second jaune, le troisième bleu. Une étiquette est collée sur chaque flacon. Alice lit l'inscription figurant sur chaque étiquette :

- Flacon rouge : le flacon jaune contient un poison ; le bleu n'en contient pas ;
- Flacon jaune : si le flacon rouge contient un poison, alors le bleu aussi ;
- Flacon bleu : je ne contiens pas de poison, mais au moins l'un des deux autres si.

Nous noterons R, J et B les variables propositionnelles correspondant au fait que les flacons rouge, jaune et bleu contiennent un poison. Nous noterons I_R, I_J et I_B les propositions correspondant aux inscriptions sur les flacons rouge, jaune et bleu.

On résoudra les questions suivantes d'abord en utilisant une table de vérité, puis en utilisant des simplifications et règles d'équivalence des formules.

(a) Exprimez les formules I_R, I_J et I_B sous la forme de formules dépendant de R, J et B.

(b) Les inscriptions sur les trois flacons sont-elles compatibles ?

(c) Dans le cas où aucun des trois flacons ne contient un poison, est-ce qu'une ou plusieurs inscriptions sont fausses ?

(d) Si les trois inscriptions sont vraies, est-ce qu'un ou plusieurs flacons contiennent un poison ?

(e) Si seuls les flacons ne contenant pas un poison ont une inscription vraie, est-ce qu'un ou plusieurs flacons ne contiennent pas un poison ?

Solution de la question (a) : I_R, I_J exprimés sous une forme dépendant de R, J et B :

$$I_R = J \wedge \neg B$$
$$I_J = R \to B$$
$$I_B = \neg B \wedge (R \vee J)$$

Solutions avec une table de vérité

On peut répondre aux questions (b) à (e) en utilisant une table de vérité.

	R	J	B	I_R	I_J	I_B	$(I_R \wedge I_J)$	$(I_J \wedge I_B)$	$(I_R \wedge I_B)$	$(I_R \wedge I_J \wedge I_B)$
1	$\bot$	$\bot$	$\bot$	$\bot$	$\top$	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$
2	$\bot$	$\bot$	$\top$	$\bot$	$\top$	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$
3	$\bot$	$\top$	$\bot$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$
4	$\bot$	$\top$	$\top$	$\bot$	$\top$	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$
5	$\top$	$\bot$	$\bot$	$\bot$	$\bot$	$\top$	$\bot$	$\bot$	$\bot$	$\bot$
6	$\top$	$\bot$	$\top$	$\bot$	$\top$	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$
7	$\top$	$\top$	$\bot$	$\top$	$\bot$	$\top$	$\bot$	$\bot$	$\top$	$\bot$
8	$\top$	$\top$	$\top$	$\bot$	$\top$	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$

Solution de la question (b): Les inscriptions sur les trois flacons sont toutes vraies si et seulement si on se trouve dans les conditions de la ligne 3, c'est-à-dire si le flacon jaune est le seul à contenir un poison.

Solution de la question (c): La situation où aucun flacon ne contient de poison correspond à la ligne 8 de la table. Alors les inscription I_R et I_B sont fausses.

Solution de la question (d): La ligne 3 correspond à la situation où les trois inscriptions sont vraies. Dans ce cas, seul le flacon jaune contient du poison ($J = \top$).

Solution de la question (e): Pour vérifier que si seuls les flacons ne contenant pas un poison ont une inscription vraie, un ou plusieurs flacons ne contiennent pas un poison, nous posons par hypothèse, que $F = (\neg R \leftrightarrow I_R) \wedge (\neg J \leftrightarrow I_J) \wedge (\neg B \leftrightarrow I_B)$ est vraie, or :

- $(\neg R \leftrightarrow I_R)$ est vraie aux lignes 3, 5, 6 et 8 ;
- $(\neg J \leftrightarrow I_J)$ est vraie aux lignes 1, 2, 6 et 7 ;
- $(\neg B \leftrightarrow I_B)$ est vraie aux lignes 2, 3, 4, 5, 6, 7 et 8.

Donc F n'est vraie qu'à la ligne 6, donc seul le flacon jaune ne contient pas de poison sous cette hypothèse.

Solutions avec des équivalences

À venir…

Chapitre 2

Logique des prédicats (du premier ordre)

Exercice 2.5.1 — Traduire dans le langage des prédicats du premier ordre les phrases suivantes :

(a) Jean suit un cours.
(b) Un logicien a été champion du monde de cyclisme.
(c) Un entier naturel est pair ou impair.
(d) Un enseignant-chercheur a toujours un nouveau sujet à étudier.
(e) Dans un triangle isocèle une médiane est également hauteur.

Dans cet exercice, on peut choisir de modéliser la phrase avec plus ou moins de précision, et plusieurs choix de modélisation sont possibles. Notez que pour faciliter la compréhension du modèle, on utilise des conventions de nommage systématiques : les prédicats unaires, qui désignent généralement des classes d'entités, sont des noms communs au singulier avec une majuscule ; les prédicats binaires sont des verbes conjugués à la troisième personne du singulier et commencent par une minuscule.

(a) On définit `Jean` comme une constante. On définit le prédicat unaire `Cours` tel que `Cours`(x) est vrai lorsque x désigne un cours ; et le prédicat binaire `suit` tel que `suit`(x,y) est vrai lorsque x suit le cours y. Alors la phrase peut s'écrire :

$\exists c\,(\texttt{Cours}(c) \wedge \texttt{suit}(\texttt{Jean},c))$

(b) On définit `cyclisme` comme une constante. On définit le prédicat unaire `Logicien` tel que `Logicien`(x) est vrai lorsque x désigne un logicien ; et le prédicat binaire `estChampionDe` tel que `estChampionDe`(x,y) est vrai lorsque x est champion dans une discipline y. Alors la phrase peut s'écrire :

$\exists l\,(\texttt{Logicien}(l) \wedge \texttt{estChampionDe}(l,\texttt{cyclisme}))$

(c) On définit le prédicat unaire `EntierNaturel` tel que `EntierNaturel`(x) est vrai lorsque x désigne un entier naturel ; le prédicat unaire `Pair` tel que `Pair`(x) est vrai lorsque x est divisible par 2 ; et le prédicat unaire `Impair` tel que `Impair`(x) est vrai lorsque x n'est divisible par 2. Alors la phrase peut s'écrire :

$\forall n\,(\texttt{EntierNaturel}(n) \rightarrow (\texttt{Pair}(n) \vee \texttt{Impair}(n)))$

Notons qu'ici, nos connaissances nous indiquent que les nombres entiers ne peuvent pas être à la fois pairs et impairs. Il serait tentant d'ajouter explicitement ce fait dans la formule, mais ce serait modéliser quelque chose qui n'est pas décrit par la phrase.

(d) Je propose d'abord une modélisation naïve de la phrase. On définit le prédicat unaire `EnseignantChercheur` tel que `EnseignantChercheur`(x) est vrai lorsque x désigne un enseignant-chercheur ; le prédicat unaire `Nouveau` tel que `Nouveau`(x) est vrai lorsque x désigne quelque chose de nouveau ; le prédicat unaire `Sujet` tel que `Sujet`(x) est vrai lorsque x désigne un sujet d'étude ; et le prédicat binaire `étudie` tel que `étudie`(x,y) est vrai lorsque x étudie le sujet y. Alors la phrase peut s'écrire :

$\forall ec\,(\texttt{EnseignantChercheur}(ec) \rightarrow \exists s\,(\texttt{Sujet}(s) \wedge \texttt{Nouveau}(s) \wedge \texttt{étudie}(ec,s)))$

Notons que ce qui est nouveau dépend du temps (une fois le nouveau sujet étudié, il n'est plus nouveau). Or, la logique du premier ordre n'est pas bien adapté pour décrire convenablement l'évolution dans le temps d'une situation. Aussi, le fait qu'un sujet soit nouveau est relatif à la personne qui le considère. Remarquons enfin que la phrase ne dit pas qu'un enseignant-chercheur étudie à tout instant un nouveau sujet. Elle indique simplement que l'enseignant-chercheur a *la possibilité* d'étudier un nouveau sujet. Là encore, la notion de potentialité ne se décrit pas facilement en logique du premier ordre. On peut néanmoins affiner la

formule pour tenter de tenir compte de ces trois remarques. On définit le prédicat unaire `Instant` tel que `Instant`(x) est vrai lorsque x désigne un instant, c'est-à-dire un point sur la ligne du temps. On définit le prédicat binaire `estEnseignantChercheur` tel que `estEnseignantChercheur`(x,y) est vrai lorsque x désigne un enseignant-chercheur à l'instant y. On définit le prédicat ternaire `estNouveauPour` tel que `estNouveauPour`(x,y,z) est vrai lorsque x désigne quelque chose de nouveau pour l'individu y à l'instant z ; le prédicat ternaire `peutÉtudier` tel que `peutÉtudier`(x,y,z) est vrai lorsque x a la possibilité d'étudier le sujet y à l'instant z. Alors la phrase peut s'écrire :

$\forall ec \forall t\,(\texttt{estEnseignantChercheur}(ec,t) \rightarrow \exists s\,(\texttt{Sujet}(s) \wedge \texttt{estNouveauPour}(s,ec,t) \wedge \texttt{peutÉtudier}(ec,s,$

(e) On définit le prédicat unaire `Triangle` tel que `Triangle`(x) est vrai lorsque x désigne un triangle ; le prédicat unaire `Isocèle` tel que `Isocèle`(x) est vrai lorsque x est isocèle ; le prédicat binaire `estMédianeDe` tel que `estMédianeDe`(x,y) est vrai lorsque x est une médiane du triangle y ; le prédicat binaire `estHauteurDe` tel que `estHauteurDe`(x,y) est vrai lorsque x est une hauteur du triangle y

$\forall t\,(\texttt{Triangle}(t) \wedge \texttt{Isocèle}(t) \rightarrow \exists m\,(\texttt{estMédianeDe}(m,t) \wedge \texttt{estHauteurDe}(m,t)))$

Notons qu'il serait erroné de définir des prédicats unaires `Médiane` et `Hauteur` car cela n'a aucun sens de parler de médiane ou de hauteur indépendemment de tout triangle.

Exercice 2.5.2 — Traduire dans le langage des prédicats du premier ordre les phrases suivantes :

(a) Tous les hommes sont méchants.
(b) Seulement les hommes sont méchants.
(c) Il existe des hommes méchants.
(d) Il existe un homme qui n'est pas méchant.
(e) Il n'existe pas d'homme méchant.
(f) Il existe un homme qui aime toutes les femmes.
(g) Chaque chat connaît un chien qui le déteste.
(h) Tous les poissons, sauf les requins, sont gentils avec les enfants.
(i) Tous les oiseaux ne peuvent pas voler.
(j) Chaque personne aime quelqu'un et personne n'aime tout le monde, ou bien quelqu'un aime tout le monde et quelqu'un n'aime personne.
(k) Il y a des gens que l'on peut rouler tout le temps et quelquefois on peut rouler tout le monde, mais on ne peut pas rouler tout le monde à chaque fois.

Dans cet exercice, on aura besoin des prédicats suivants, dont la signification est sensée être claire de par le choix de leur nom, sauf exceptions détaillées entre parenthèses. On utilise les mêmes conventions que pour l'exercice 2.5.1 :

Prédicats unaires :

`Homme`, `Méchant`, `Femme`, `Chat`, `Chien`, `Poisson`, `Requin`, `Enfant`, `Oiseau`, `Volant` (quelque chose qui peut voler)

Prédicats binaires :

`aime`, `connait`, `déteste`, `estGentilAvec`, `peutÊtreRoulé` (vrai lorsque le premier argument peut être roulé, dans le sens de trompé/dupé, à l'instant du second argument)

(a) $\forall x\,(\texttt{Homme}(x) \rightarrow \texttt{Méchant}(x))$
(b) $\forall x\,(\texttt{Méchant}(x) \rightarrow \texttt{Homme}(x))$
(c) $\exists x\,(\texttt{Homme}(x) \wedge \texttt{Méchant}(x))$[1]
(d) $\exists x\,(\texttt{Homme}(x) \wedge \neg\texttt{Méchant}(x))$
(e) $\neg\exists x\,(\texttt{Homme}(x) \wedge \texttt{Méchant}(x))$
(f) $\exists x\,(\texttt{Homme}(x) \wedge \forall y\,(\texttt{Femme}(y) \rightarrow \texttt{aime}(x,y)))$
(g) $\forall x\,(\texttt{Chat}(x) \rightarrow > \exists y\,(\texttt{Chien}(y) \wedge \texttt{connait}(x,y) \wedge \texttt{déteste}(y,x)))$
(h) $\forall x\,((\texttt{Poisson}(x) \wedge \neg\texttt{Requin}(x)) \rightarrow \forall y\,(\texttt{Enfant}(y) \rightarrow \texttt{estGentilAvec}(x,y)))$[2]
(i) $\neg\forall x\,(\texttt{Oiseau}(x) \rightarrow \texttt{Volant}(x))$

(j) $((\forall x \exists y\ \texttt{aime}(x,y) \land \neg\exists x \forall y\ \texttt{aime}(x,y)) \lor (\exists x \forall y\ \texttt{aime}(x,y) \land \exists x \forall y\ \neg\texttt{aime}(x,y)))$

(k) $(\exists x \forall t\ \texttt{peutÊtreRoulé}(x,t) \land \exists t \forall x\ \texttt{peutÊtreRoulé}(x,t) \land \forall x \forall t\ \neg\texttt{peutÊtreRoulé}(x,t))$

1. On notera que la phrase utilise un pluriel, alors que la formule ne mentionne l'existence que d'une seule entité. On pourrait penser que le pluriel peut s'écrire $\exists x \exists y \ldots$ mais cela nindique pas que x et y désigne des choses différentes. Il faudrait pouvoir indiquer que $x \neq y$ mais le symbole $\neq$ n'appartient pas au langage de la logique du premier ordre. Une manière de s'en sortir consiste à introduire un prédicat unaire `EstX` et d'écrire $\exists x \exists y\ (\texttt{Homme}(x) \land \texttt{Méchant}(x) \land \texttt{EstX}(x) \land \texttt{Homme}(y) \land \texttt{Méchant}(y) \land \neg\texttt{EstX}(y))$. Ceci impose nécessairement que x est différent de y.

2. La formule, telle qu'écrite, dit que « tous les poissons qui ne sont pas des requins sont gentils avec tous les enfants ». Ceci ne dit rien sur les requins. En revanche, le mot « sauf » laisse penser quil y a des exceptions (ou au moins une) parmi les requins. Par analogie, si on disait « tous les nombres entiers naturels, sauf le nombre zéro, ont un prédécesseur », on comprendrait que zéro est une exception. Cela na pas le même sens que « tous les entiers naturels non nuls ont un prédesseur », car on pourrait également dire « tous les entiers naturels non nuls ont un successeur », alors quil serait incorrect de dire « tous les nombres entiers naturels, sauf zéro, ont un successeur ». On peut modéliser lexception en ajoutant $\exists x \exists y\ (\texttt{Requin}(x) \land \neg\texttt{estGentilAvec}(x,y))$

Exercice 2.5.3 — Traduire dans le langage des prédicats du premier ordre les phrases suivantes :

(a) Toutes les personnes qui entrent en voiture dans la faculté doivent avoir une carte ou être accompagnées par un membre du personnel.

(b) Certains étudiants entrent en voiture dans la faculté sans être accompagnés de personnes qui ne sont pas des étudiants.

(c) Aucun étudiant n'a de carte.

On peut proposer différent niveaux de précisions dans cette modélisation. Dans un premier temps, on ignore l'aspect temporel et on simplifie quelque peu la situation avec des prédicats regroupant plusieurs caractéristiques :

Prédicats unaires :

`IndividuEntrantDansLaFac`, `PossesseurDeCarte`, `Étudiant`, `MembreDuPersonnel`

Prédicats binaires :

`accompagne`

(a) $\forall x\ (\texttt{IndividuEntrantDansLaFac}(x) \rightarrow (\texttt{PossesseurDeCarte}(x,c) \lor \exists p\ (\texttt{MembreDuPersonnel}(p) \land \texttt{accompagne}(p,x))))$

(b) $\exists x\ (\texttt{Étudiant}(x) \land \texttt{IndividuEntrantDansLaFac}(x) \land \neg\exists y\ (\texttt{accompagne}(x,y) \land \neg\texttt{Étudiant}(y)))$

(c) $\forall x\ (\texttt{Étudiant}(x) \rightarrow \neg\texttt{PossesseurDeCarte}(x))$

Notons que ceci permet de conclure que certains étudiants sont membres du personnel (pouvez-vous le prouver ?). On peut affiner, toujours sans tenir compte de l'aspect temporel :

Constante :

`laFaculté`

Prédicats unaires :

`Personne`, `Voiture`, `Carte`

Prédicats binaires :

`entreDans`, `possède`, `conduit`

(a) $\forall x \forall v\ ((\texttt{Personne}(x) \land \texttt{Voiture}(v) \land \texttt{conduit}(x,v) \land \texttt{entreDans}(x,\texttt{faculté})) \rightarrow (\exists c\ (\texttt{Carte}(c) \land \texttt{possède}(x,c)) \lor \exists p\ (\texttt{MembreDuPersonnel}(p) \land \texttt{accompagne}(p,x))))$

(b) $\exists x \exists v\ (\texttt{Étudiant}(x) \land \texttt{Voiture}(v) \land \texttt{conduit}(x,v) \land \texttt{entreDans}(x,\texttt{faculté}) \land \neg\exists y\ (\texttt{accompagne}(x,y) \land \neg\texttt{Étudiant}(y)))$

(c) $\forall x\ (\texttt{Étudiant}(x) \rightarrow \neg\exists c\ (\texttt{Carte}(c) \land \texttt{possède}(x,c)))$

Notons que ceci *ne permet pas* de conclure que certains étudiants sont membres du personnel (on peut construire une interprétation qui satisfait ces formules et dans laquelle aucun étudiant

n'est membre du personnel). Il peut sembler surprenant qu'en affinant le modèle, on perde certaines conclusions. La raison est que dans la première des trois formules, la condition impose que x désigne une personne (car la phrase disait « toutes les personnes qui… ») alors que les deux autres formules ne parlent pas a priori de personnes ! En effet, rien dans la formule ne permet de conclure qu'un étudiant est une personne. On pourrait relâcher la contrainte de la première formule en éliminant $\texttt{Personne}(x)$ (toute entité qui entre en voiture dans la faculté est soumise à la contrainte) ou bien modéliser le fait qu'un étudiant est une personne :

(a) $\forall x(\texttt{Étudiant}(x) \rightarrow \texttt{Personne}(x))$

Enfin, on peut ajouter la dimension temporel. En effet, pour que la première formule décrire bien la réalité, il faut qu'elle soit vraie à tout instant, tandis que la seconde formule ne doit être vraie qu'à certains moments. On peut donc introduire le prédicat unaire `Instant` et transformer les prédicats `entreDans`, `conduit`, `possède` et `accompagne` en prédicats ternaires. Ainsi :

(a) $\forall t \forall x \forall v\, ((\texttt{Instant}(t) \wedge \texttt{Voiture}(v) \wedge \texttt{conduit}(x,v,t) \wedge \texttt{entreDans}(x,\texttt{faculté},t)) \rightarrow (\exists c\, (\texttt{Carte}(c) \wedge \texttt{possède}(x,c,t)) \vee \exists p\, (\texttt{MembreDuPersonnel}(p) \wedge \texttt{accompagne}(p,x,t))))$
(b) $\exists t \exists x \exists v\, (\texttt{Instant}(t) \wedge \texttt{Étudiant}(x) \wedge \texttt{Voiture}(v) \wedge \texttt{conduit}(x,v,t) \wedge \texttt{entreDans}(x,\texttt{faculté},t) \wedge \neg\exists y\, (\texttt{accompagne}(x,y,t) \wedge \neg\texttt{Étudiant}(y)))$
(c) $\forall t \forall x\, ((\texttt{Instant}(t) \wedge \texttt{Étudiant}(x)) \rightarrow \neg\exists c\, (\texttt{Carte}(c) \wedge \texttt{possède}(x,c,t)))$

Exercice 2.5.4 — Traduire dans le langage des prédicats du premier ordre les phrases suivantes :

> *Une conjecture est un théorème qui ne peut être démontré par aucun mathématicien. Il existe des mathématiciens qui ne démontrent pas tous les théorèmes. Si un mathématicien démontre une conjecture alors il se trompe. Si quelqu'un démontre un théorème sans se tromper, alors ce n'est pas une conjecture.*

On introduit les prédicats suivants :

Prédicats unaires :
`Conjecture`, `Mathématicien`, `Théorème`, `EnErreur` (quelqu'un qui se trompe)
Prédicats binaires :
`démontre`

Ainsi, les phrases peuvent se modéliser comme suit :

(i) $\forall c \forall m\, ((\texttt{Conjecture}(c) \wedge \texttt{Mathématicien}(m)) \rightarrow (\texttt{Théorème}(c) \wedge \neg\texttt{démontre}(m,c)))$
(ii) $\exists m \exists t\, (\texttt{Mathématicien}(m) \wedge \texttt{Théorème}(t) \wedge \neg\texttt{démontre}(m,t))$
(iii) $\forall m \forall c\, ((\texttt{Mathématicien}(c) \wedge \texttt{Conjecture}(m) \wedge \texttt{démontre}(m,c)) \rightarrow \texttt{EnErreur}(m))$
(iv) $\forall x \forall t\, ((\texttt{Théorème}(t) \wedge \neg\texttt{EnErreur}(x)) \rightarrow \neg\texttt{Conjecture}(t))$

Exercice 2.5.5 — Traduire dans le langage des prédicats du premier ordre les phrases suivantes :

> *Il existe des PC non connectés en réseau. Dans les grandes entreprises, tous les PC sont connectés au réseau interne. Il existe dans chaque grande entreprise au moins un PC connecté au réseau interne et relié à Internet.*

On introduit les termes suivants :

Constante :
`Internet`
Prédicats unaires :
`GrandeEntreprise`, `PC`, `Réseau`

Prédicats binaires :

`estConnectéÀ`, `possède`, `seTrouveDans`

Ainsi, les phrases peuvent se modéliser comme suit :

(i) $\exists x \forall y\, (\texttt{PC}(x) \wedge (\texttt{Réseau}(y) \rightarrow \neg\texttt{estConnectéÀ}(x,y)))$
(ii) $\forall x \forall y\, ((\texttt{GrandeEntreprise}(x) \wedge \texttt{PC}(y) \wedge \texttt{Réseau}(z) \wedge \texttt{seTrouveDans}(y,x) \wedge \texttt{seTrouveDans}(z,x)) \rightarrow \texttt{estConnectéÀ}(y,z))$[3]
(iii) $\forall x \forall y \exists z\, ((\texttt{GrandeEntreprise}(x) \wedge \texttt{Réseau}(y) \wedge \texttt{seTrouveDans}(y,x)) \rightarrow (\texttt{PC}(z) \wedge \texttt{seTrouveDans}(z,x) \wedge \texttt{estConnectéÀ}(z,y) \wedge \texttt{estConnectéÀ}(z,\texttt{Internet})))$[3]

3. On modélise « le réseau interne d'une entreprise » comme un réseau qui se trouve dans l'entreprise.

Exercice 2.5.6 — Traduire dans le langage des prédicats du premier ordre les phrases suivantes :

> *Toute personne possédant un PC et un modem peut utiliser Internet. Il existe des personnes connectées en réseau, qui ne peuvent pas utiliser Internet. Nul ne possède de modem s'il ne possède pas de PC. Tout utilisateur d'Internet est soit connecté à un réseau, soit possesseur d'un PC. Il n'est pas possible d'utiliser Internet sans posséder de PC.*

On introduit les termes suivants :

Constante :

`Internet`

Prédicats unaires :

`Modem`, `PC`, `Personne`, `Réseau`

Prédicats binaires :

`estConnectéÀ`, `possède`

Ainsi, les phrases peuvent se modéliser comme suit :

(i) $\forall x \forall y \forall z\, ((\texttt{Personne}(x) \wedge \texttt{PC}(y) \wedge \texttt{Modem}(z) \wedge \texttt{possède}(x,y) \wedge \texttt{possède}(x,z)) \rightarrow \text{estConnectéÀ}(x,\texttt{Internet}))$[4]
(ii) $\exists x \exists y\, ((\texttt{Personne}(x) \wedge \texttt{Réseau}(y) \wedge \text{estConnectéÀ}(x,y) \wedge \neg\text{estConnectéÀ}(x,\texttt{Internet}))$
(iii) $\forall x \forall y\, ((\texttt{Modem}(y) \wedge \texttt{possède}(x,y)) \rightarrow \exists z(\text{PC}(z) \wedge \texttt{possède}(x,z)))$
(iv) $\forall x\, (\text{estConnectéÀ}(x,\texttt{Internet}) \rightarrow (\exists y\, (\texttt{Réseau}(y) \wedge \text{estConnectéÀ}(x,y)) \vee \exists z\, (\texttt{PC}(z) \wedge \texttt{possède}(x,z))))$
(v) $\forall x\, (\text{estConnectéÀ}(x,\texttt{Internet}) \rightarrow \exists y\, (\texttt{PC}(y) \wedge \texttt{possède}(x,y)))$

Une version simplifiée acceptable pourrait utiliser les prédicats suivants :

Prédicats unaires :

`EnRéseau`, `PossesseurDeModem`, `PossesseurDePC`, `Internaute` (peut se connecter à Internet)

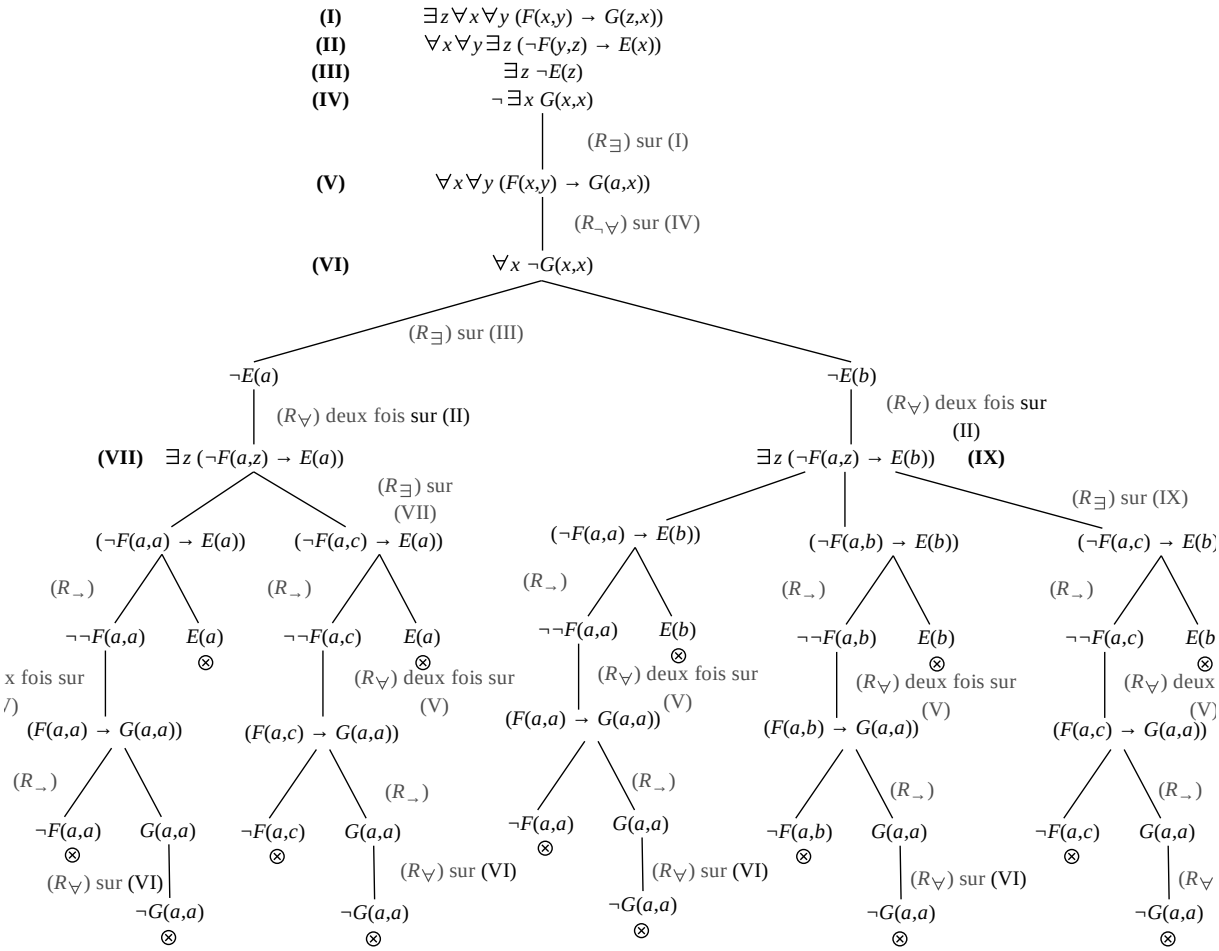
(i) $\forall x\, ((\texttt{PossesseurDePC}(x) \wedge \texttt{PossesseurDeModem}(x)) \rightarrow \text{Internaute}(x))$
(ii) $\exists x \texttt{EnRéseau}(x) \wedge \neg\text{Internaute}(x))$
(iii) $\forall x\, (\texttt{PossesseurDeModem}(x) \rightarrow \texttt{PossesseurDePC}(x))$
(iv) $\forall x\, (\text{Internaute}(x) \rightarrow (\texttt{EnRéseau}(x) \vee \texttt{PossesseurDePC}(x)))$
(v) $\forall x\, (\text{Internaute}(x) \rightarrow \texttt{PossesseurDePC}(x))$

4. On considèrera que « peut utiliser Internet » signifie la même chose que « est connecté à Internet ».

Exercice 2.5.7 — Soient les expressions dans $\mathscr{L}_1$:

(A_1)	$\exists z \forall x \forall y\, (F(x,y) \to G(z,x))$
(A_2)	$\forall x \forall y \exists z\, (\neg F(y,z) \to E(x))$
(A_3)	$\exists z\, \neg E(z)$
(C)	$\exists x\, G(x,x)$

Montrer, en utilisant la méthode des tableaux, que $A_1 \wedge A_2 \wedge A_3 \to C$ est un théorème.



Arbre de décision pour l'exercice 2.5.7

Chapitre 3

Principe de résolution

Exercice 3.5.1 — Montrer, en utilisant la résolution, que la formule suivante est une tautologie :

$$((p \to r) \wedge (q \leftrightarrow s) \wedge (r \leftrightarrow q) \wedge p) \to s$$

Il faut montrer que $F = ((p \to r) \wedge (q \leftrightarrow s) \wedge (r \leftrightarrow q) \wedge p) \wedge \neg s$ est une contradiction. Comme il sagit dune formule de logique propositionnelle, il ny a pas de variable, donc il nest pas nécessaire de mettre sous forme prénexe ni sous forme de Skolem.

On obtient la forme normale conjonctive :

$$\begin{aligned} F &= ((p \to r) \wedge (q \leftrightarrow s) \wedge (r \leftrightarrow q) \wedge p) \wedge \neg s \\ &\equiv (p \to r) \wedge (q \to s) \wedge (s \to q) \wedge (r \to q) \wedge (q \to r) \wedge p \wedge \neg s \\ &\equiv (\neg p \vee r) \wedge (\neg q \vee s) \wedge (\neg s \vee q) \wedge (\neg r \vee q) \wedge (\neg q \vee r) \wedge p \wedge \neg s \end{aligned}$$

Puis en forme clausale :

$$\left|\begin{array}{ll} (C_1) & \neg p \quad r \\ (C_2) & \neg q \quad s \\ (C_3) & \neg s \quad q \\ (C_4) & \neg r \quad q \\ (C_5) & \neg q \quad r \\ (C_6) & p \\ (C_7) & \neg s \end{array}\right.$$

On peut obtenir la clause vide grâce à la suite de clauses :

$$\left|\begin{array}{lll} (C_8) & \neg q & \text{obtenue par la résolution de } C_2 + C_7 \\ (C_9) & \neg r & \text{obtenue par la résolution de } C_4 + C_8 \\ (C_{10}) & \neg p & \text{obtenue par la résolution de } C_1 + C_9 \\ (C_{11}) & \square & \text{obtenue par la résolution de } C_6 + C_{10} \end{array}\right.$$

Exercice 3.5.2 — Soient les formules dans $\mathscr{L}_1$:

$$\begin{array}{ll} (A_1) & \exists z \forall x \forall y\, (F(x,y) \to G(z,x)) \\ (A_2) & \forall x \forall y \exists z\, (\neg F(y,z) \to E(x)) \\ (A_3) & \exists z\, \neg E(z) \\ (C) & \exists x\, G(x,x) \end{array}$$

Montrer, en utilisant le principe de résolution, que $A_1 \wedge A_2 \wedge A_3 \to C$ est un théorème.

Il faut montrer que $A_1 \wedge A_2 \wedge A_3 \wedge \neg C$ est contradictoire. Mise sous forme prénexe :

$$\begin{array}{ll} (A_1) & \exists z \forall x \forall y\, (F(x,y) \to G(z,x)) \\ (A_2) & \forall x \forall y \exists z\, (\neg F(y,z) \to E(x)) \\ (A_3) & \exists z\, \neg E(z) \\ (\neg C) & \forall x\, \neg G(x,x) \end{array}$$

Mise sous forme de Skolem (a et b sont des constantes) :

(A_1)	$(F(x,y) \to G(a,x))$
(A_2)	$(\neg F(y,f(x,y)) \to E(x))$
(A_3)	$\neg E(b)$
$(\neg C)$	$\neg G(x,x)$

Mise sous forme normale conjonctive :

(A_1)	$(\neg F(x,y) \vee G(a,x))$
(A_2)	$(F(y,f(x,y)) \vee E(x))$
(A_3)	$\neg E(b)$
$(\neg C)$	$\neg G(x,x)$

Mise sous forme clausale et renommage des variables pour éviter des conflits lors de l'unification :

(C_1)	$\neg F(x_1,y_1)$	$G(a,x_1)$
(C_2)	$F(y_2,f(x_2,y_2)$	$E(x_2)$
(C_3)	$\neg E(b)$	
(C_4)	$\neg G(x_4,x_4)$	

On applique alors la résolution de la façon suivante :

(C_5)	$\neg F(a,y_1)$	obtenue par la résolution de $C_1 + C_4$	$[x_4 \mapsto a, x_1 \mapsto a]$
(C_6)	$E(x_2)$	obtenue par la résolution de $C_2 + C_5$	$[y_2 \mapsto a, y_1 \mapsto f(x_2,a)]$
(C_7)	⬜	obtenue par la résolution de $C_3 + C_6$	$[x_2 \mapsto b]$

Exercice 3.5.3 — Montrer en utilisant le principe de résolution que :

$$H_1, H_2, H_3 \models C$$

On prendra soin de mettre en valeur les étapes de mise sous forme prénexe, de Skolémisation et de mise sous FNC.

(H_1)	$(\exists x\, F(x) \to \exists x\, G(x))$
(H_2)	$\forall x\, (G(x) \to \exists y\, H(x,y))$
(H_3)	$\forall x \forall y\, (H(x,y) \to A(x))$
(C)	$\exists x\, (F(x) \to A(x))$

Il faut montrer que $H_1 \wedge H_2 \wedge H_3 \wedge \neg C$ est contradictoire. Mise sous forme prénexe :

(H_1)	$\forall x \exists y\, (F(x) \to G(y))$
(H_2)	$\forall x \exists y\, (G(x) \to H(x,y))$
(H_3)	$\forall x \forall y\, (H(x,y) \to A(x))$
$(\neg C)$	$\forall x\, \neg(F(x) \to A(x))$

Notons qu'il faut renommer une des deux occurrences de la variable x dans la formule (H_1) car elles sont quantifiées séparément. Mise sous forme de Skolem :

(H_1)	$(F(x) \to G(f(x)))$
(H_2)	$(G(x) \to H(x,g(x)))$
(H_3)	$(H(x,y) \to A(x))$
$(\neg C)$	$\neg(F(x) \to A(x))$

Notons qu'il faut introduire un symbole de fonction différent pour chaque occurrence d'un quantificateur existentiel. Mise sous forme normale conjonctive :

(H_1) $(\neg F(x) \lor G(f(x)))$
(H_2) $(\neg G(x) \lor H(x,g(x)))$
(H_3) $(\neg H(x,y) \lor A(x))$
$(\neg C)$ $(F(x) \land \neg A(x))$

Mise sous forme clausale et renommage des variables pour éviter des conflits lors de l'unification :

(C_1)	$\neg F(x_1)$	$G(f(x_1))$
(C_2)	$\neg G(x_2)$	$H(x_2,g(x_2))$
(C_3)	$\neg H(x_3,y_3)$	$A(x_3)$
(C_4)	$F(x_4)$	
(C_5)	$\neg A(x_5)$	

Notons qu'il y a une conjonction dans la formule $(\neg C)$, ce qui aboutit à deux clauses qu'il faut disposer sur deux lignes différentes. On applique alors la résolution de la façon suivante :

(C_6)	$\neg H(x_3,y_3)$	$(C_3 + C_5)$	$[x_4 \mapsto x_3]$
(C_7)	$\neg G(x_2)$	$(C_2 + C_6)$	$[x_3 \mapsto x_2, y_3 \mapsto g(x_2)]$
(C_8)	$\neg F(x_1)$	$(C_1 + C_7)$	$[x_2 \mapsto f(x_1)]$
(C_9)	⬜	$(C_4 + C_8)$	$[x_1 \mapsto x_4]$

Exercice 3.5.4 — Montrer en utilisant le principe de résolution que :

$H_1, H_2, H_3, H_4, H_5 \vDash C$

On prendra soin de mettre en valeur les étapes de mise sous forme prénexe, de Skolémisation et de mise sous FNC. La solution devra emprunter *toutes les clauses de départ.*

(H_1) $\forall x \forall y\, (M(x,x,x) \lor B(x))$
(H_2) $\forall x \forall y \exists z\, ((M(x,y,z) \land N(x,y)) \to P(y,x))$
(H_3) $\exists x \forall y \forall z \forall t\, (\neg P(y,x) \land (M(y,z,t) \lor A(x,z)))$
(H_4) $\forall x \forall y\, (M(x,x,x) \to N(x,y))$
(H_5) $\forall x \forall y\, (M(x,x,y) \to \neg A(x,y))$
(C) $\exists x\, B(x)$

(x, y, z et t désignent des variables)

Il faut montrer que $H_1 \land H_2 \land H_3 \land H_4 \land H_5 \land \neg C$ est contradictoire. Mise sous forme prénexe :

(H_1) $\forall x \forall y\, (M(x,x,x) \lor B(x))$
(H_2) $\forall x \forall y \exists z\, ((M(x,y,z) \land N(x,y)) \to P(y,x))$
(H_3) $\exists x \forall y \forall z \forall t\, (\neg P(y,x) \land (M(y,z,t) \lor A(x,z)))$
(H_4) $\forall x \forall y\, (M(x,x,x) \to N(x,y))$
(H_5) $\forall x \forall y\, (M(x,x,y) \to \neg A(x,y))$
$(\neg C)$ $\forall x\, \neg B(x)$

Mise sous forme de Skolem (a désigne une constante) :

(H_1) $(M(x,x,x) \lor B(x))$
(H_2) $((M(x,y,f(x,y)) \land N(x,y)) \to P(y,x))$

(H_3) $(\neg P(y,a) \wedge (M(y,z,t) \vee A(a,z)))$
(H_4) $(M(x,x,x) \to N(x,y))$
(H_5) $(M(x,x,y) \to \neg A(x,y))$
$(\neg C)$ $\neg B(x)$

Mise sous forme normale conjonctive :

(H_1) $(M(x,x,x) \vee B(x))$
(H_2) $(\neg M(x,y,f(x,y)) \vee \neg N(x,y) \vee P(y,x))$
(H_3) $(\neg P(y,a) \wedge (M(y,z,t) \vee A(a,z)))$
(H_4) $(\neg M(x,x,x) \vee N(x,y))$
(H_5) $(\neg M(x,x,y) \vee \neg A(x,y))$
$(\neg C)$ $\neg B(x)$

Mise sous forme clausale et renommage des variables pour éviter des conflits lors de l'unification ((H_3) donne lieu à deux clauses distinctes) :

(C_1)	$M(x_1,x_1,x_1)$	$B(x_1)$	
(C_2)	$\neg M(x_2,y_2,f(x_2,y_2))$	$\neg N(x_2,y_2)$	$P(y_2,x_2)$
(C_3)	$\neg P(y_3,a)$		
(C_4)	$M(y_4,z_4,t_4)$	$A(a,z_4)$	
(C_5)	$\neg M(x_5,x_5,x_5)$	$N(x_5,y_5)$	
(C_6)	$\neg M(x_6,x_6,y_6)$	$\neg A(x_6,y_6)$	
(C_7)	$\neg B(x_7)$		

On applique alors la résolution de la façon suivante :

(C_8)	$M(x_7,x_7,x_7)$		$(C_1 + C_7)$	$[x_1 \mapsto x_7]$
(C_9)	$\neg A(x_7,x_7)$		$(C_6 + C_8)$	$[x_6 \mapsto x_7, y_6 \mapsto x_7]$
(C_{10})	$M(y_4,a,t_4)$		$(C_4 + C_9)$	$[x_7 \mapsto a, z_4 \mapsto a]$
(C_{11})	$\neg N(x_2,a)$	$P(a,x_2)$	$(C_2 + C_{10})$	$[y_4 \mapsto x_2, y_2 \mapsto a, t_4 \mapsto f(x_2,a)]$
(C_{12})	$\neg N(a,a)$		$(C_3 + C_{11})$	$[x_2 \mapsto a, y_3 \mapsto a]$
(C_{13})	$\neg M(a,a,a)$		$(C_5 + C_{12})$	$[x_5 \mapsto a, y_5 \mapsto a]$
(C_{14})	⬜		$(C_8 + C_{13})$	$[x_7 \mapsto a]$

Exercice 3.5.5 — Soient les formules suivantes dans $\mathscr{L}_1$. Montrer, en utilisant le principe de résolution, que $H_1 \wedge H_2 \wedge H_3 \wedge H_4 \to C$ est un théorème. La solution devra emprunter toutes les clauses de départ. x, y, z désignent des variables.

(H_1) $\forall x \forall y\, (D(x,y) \to A(x,x))$
(H_2) $(\forall x\, A(x,x) \to \forall x \forall y\, B(x,y))$
(H_3) $\forall x \forall y \forall z\, (C(x,z) \to (A(x,y) \vee D(y,z) \vee E(z,z)))$
(H_4) $\forall x \exists y\, (C(x,y) \wedge \neg E(y,y))$
(C) $\exists x\, B(x,x)$

Il faut montrer que $H_1 \wedge H_2 \wedge H_3 \wedge H_4 \wedge \neg C$ est contradictoire. Mise sous forme prénexe :

(H_1) $\forall x \forall y\, (D(x,y) \to A(x,x))$
(H_2) $\exists z \forall x \forall y\, (A(z,z) \to B(x,y))$
(H_3) $\forall x \forall y \forall z\, (C(x,z) \to (A(x,y) \vee D(y,z) \vee E(z,z)))$
(H_4) $\forall x \exists y\, (C(x,y) \wedge \neg E(y,y))$

$(\neg C)$ $\forall x\, \neg B(x,x)$

Mise sous forme de Skolem (a désigne une constante) :

(H_1) $(D(x,y) \to A(x,x))$
(H_2) $(A(a,a) \to B(x,y))$
(H_3) $(C(x,z) \to (A(x,y) \lor D(y,z) \lor E(z,z)))$
(H_4) $(C(x,f(x)) \land \neg E(f(x),f(x)))$
$(\neg C)$ $\neg B(x,x)$

Mise sous forme normale conjonctive :

(H_1) $(\neg D(x,y) \lor A(x,x))$
(H_2) $(\neg A(a,a) \lor B(x,y))$
(H_3) $(\neg C(x,z) \lor A(x,y) \lor D(y,z) \lor E(z,z))$
(H_4) $(C(x,f(x)) \land \neg E(f(x),f(x)))$
$(\neg C)$ $\neg B(x,x)$

Mise sous forme clausale et renommage des variables pour éviter des conflits lors de l'unification ((H_4) donne lieu à deux clauses distinctes) :

(C_1)	$\neg D(x_1,y_1)$	$A(x_1,x_1)$		
(C_2)	$\neg A(a,a)$	$B(x_2,y_2)$		
(C_3)	$\neg C(x_3,z_3)$	$A(x_3,y_3)$	$D(y_3,z_3)$	$E(z_3,z_3)$
(C_4)	$C(x_4,f(x_4))$			
(C_5)	$\neg E(f(x_5),f(x_5))$			
(C_6)	$\neg B(x_6,x_6)$			

On applique alors la résolution de la façon suivante :

(C_7)	$\neg A(a,a)$		$(C_2 + C_6)$	$[x_2 \mapsto x_6, y_2 \mapsto x_6]$
(C_8)	$\neg D(a,y_1)$		$(C_1 + C_7)$	$[x_1 \mapsto a]$
(C_9)	$\neg C(x_3,z_3)$	$A(x_3,a)$ $E(z_3,z_3)$	$(C_3 + C_8)$	$[y_3 \mapsto a, y_1 \mapsto z_3]$
(C_{10})	$\neg C(x_3,f(x_5))$	$A(x_3,a)$	$(C_5 + C_9)$	$[z_3 \mapsto f(x_5)]$
(C_{11})	$A(x_3,a)$		$(C_4 + C_{10})$	$[x_4 \mapsto x_3, x_5 \mapsto x_3]$
(C_{12})	⬜		$(C_7 + C_{11})$	$[x_3 \mapsto a]$

Exercice 3.5.6 — En reprenant le monde de l'exercice 2.5.3, montrer que certains étudiants sont membres du personnel.

Todo...

Université Bretagne Sud
L3 Informatique

Logique

Travaux Dirigés - Partie 7
Corrigés

Ce septième TD est consacré à des compléments pour la Logique Propositionnelle.

Les exercices sont de difficultés diverses et sont à traiter en se basant sur les notions introduites en cours (chapitre 3) et en annexe de ce TD.

J'ai choisi de présenter **la méthode de résolution** *pour la LP0 dans l'annexe 1 du TD pour ne pas surcharger le cours.*

L'annexe 2 contient des développement autour de la notion de **stratégie** *(de preuve).*

Il est nécessaire d'étudier ces annexes avant d'aborder les exercices du TD.

Rappelons qu'on appelle **clause** *une disjonction de littéraux de la LP0 et que si tous les littéraux d'une clause sont positifs (respectivement négatifs) la clause est dite* **positive** *(respectivement* **négative***).*
Étant donné un ensemble de clauses, un littéral apparaissant dans une clause et dont la négation n'apparaît pas dans les autres clauses, est appellé un **littéral pur.**

Bon travail !

Exercice 1

Prouver la complétude pour la réfutation de la méthode de résolution.

Il faut prouver :

$$\text{Si } S \text{ insat, alors } S \vdash_R \square$$

On fera une preuve par récurrence sur Je nombre de littéraux positifs de S.

i) $n = 1$ (si $n = 0$, S est satisfaisable).

Comme S insat alors S est nécessairement de la forme :

$$S = \{L, L^c\}$$

donc en appliquant la résolution une fois on obtient :

$$S \vdash_R \square$$

ii) $n > 1$

L'idée est d'appliquer une transformation préservant l'insatisfaisabilité, mais faisant décroître strictement le nombre de littéraux (pour pouvoir appliquer l'hypothèse de récurrence).

On suppose que S contient $n + 1$ littéraux.

Choisir un littéral non pur $L \in C \in S$.

Considérer les ensembles de clauses :

$$S_1 = \{C \setminus \{L\} | C \in S \text{ et } L^c \notin C\}$$

$$S_2 = \{C \setminus \{L^c\} | C \in S \text{ et } L \notin C\}$$

Alors, S est insat si et seulement si S_1 insat et S_2 insat (cf. TD 6 – exercice 2, correction et complétude de la méthode de Davis et Putnam).

Par hypothèse de récurrence, $S_1 \vdash_R \square$ (S_1 contient au plus n littéraux).

Deux cas sont à analyser (le deuxième à son tour donne lieu à deux cas) :

a) les clauses $C \setminus \{L\}$ n'interviennent pas dans la réfutation de S_1.

Les clauses qui interviennent dans la réfutation sont aussi dans S, donc :

$$S \vdash_R \square$$

b1) les clauses $C \setminus \{L\}$ interviennent dans la réfutation de S_1.

Dans ce cas $S \vdash_R L$, comme l'on peut s'en convaincre avec l'exemple suivant :

1. $A \vee B$
2. $\neg A \vee B$

3. $A \vee \neg B$
4. $\neg A \vee \neg B$

5. B	$(1,1) - (2,1)$
6. $\neg B$	$(3,1) - (4,1)$
7. $\square$	$(5,1) - (6,1)$

Si l'on ajoute à la première clause un littéral pur :

1. $A \vee B \vee \boxed{\mathbf{L}}$
2. $\neg A \vee B$
3. $A \vee \neg B$
4. $\neg A \vee \neg B$

5. $B \vee \boxed{\mathbf{L}}$	$(1,1) - (2,1)$
6. $\neg B$	$(3,1) - (4,1)$
7. $\boxed{\mathbf{L}}$	$(5,1) - (6,1)$

b2) en répétant exactement le même raisonnement pour S_2 avec L^c à la place de L, on conclut à :

$$S \vdash_R L^c$$

Maintenant pour S (contenant $n+1$ littéraux) on a :

$$S \vdash_R L \text{ et } S \vdash_R L^c$$

donc en appliquant une fois la règle de résolution, on obtient : $S \vdash_R \square$.

Exercice 2

Prouver que dans une réfutation par résolution l'on peut éliminer les tautologies sans perdre la complétude réfutationnelle.

On doit montrer :

$$\text{si } S \vdash_R \square \text{ alors } S \vdash_{R+et} \square$$

où *et* désigne l'élimination des tautologies.

On montre d'abord que si S insat alors $\mathcal{R}^n(S)$ $(n \geq 1)$ insat.

La preuve en est triviale : tout sur-ensemble d'un ensemble insat est insat (voir exercice 10).

Comme l'on a montré la complétude pour la réfutation de la résolution, il suffit d'appliquer la règle R-0 de Davis et Putnam à $\mathcal{R}^n(S)$ (ensembles de clauses insatisfaisables) pour chaque $n \geq 1$.

Exercice 3

Comment détecter par résolution qu'un ensemble de clauses est satisfaisable ?

Donner un exemple.

Puisque la résolution est complète pour la réfutation, si S insat alors, après un nombre fini de pas, $\square$ sera produite.

Étant donné un ensemble de clauses S, le problème est de décider si S est sat ou insat.

Comme S contient un nombre fini de littéraux et l'application de R_c n'en ajoute pas, après un nombre fini de pas, soit on a produit $\square$, soit on ne produit pas de nouvelle clause.

Dans ce dernier cas on dit que l'on est arrivé à *saturation*.

Quand on arrive à la saturation de l'ensemble de conséquences de S, la complétude de la résolution pour la réfutation nous permet de conclure à la satisfaisabilité de S.

Un exemple de l'application de cette procédure de décision est la détection de la satisfaisabilité de l'ensemble de clauses 1., 2., 3. ci-dessous :

1. $A \vee B$
2. $\neg A \vee B$
3. $\neg A \vee \neg B$
4. B $\quad (1,1) - (2,1)$
5. $B \vee \neg B$ $\quad (1,1) - (3,1)$
6. $A \vee \neg A$ $\quad (1,2) - (3,2)$
7. $\neg A$ $\quad (2,2) - (3,2)$

Exercice 4

Utiliser la méthode de résolution pour répondre aux questions suivantes.

a) Prouver que $\mathcal{S}$ est insatisfaisable :

$$\mathcal{S} = \{P, \neg P \vee Q, \neg Q \vee R, \neg Q \vee \neg R\}$$

1. P
2. $\neg P \vee Q$
3. $\neg Q \vee R$
4. $\neg Q \vee \neg R$
5. Q $\quad (1,1) - (2,1)$
6. R $\quad (5,1) - (3,1)$
7. $\neg R$ $\quad (5,1) - (4,1)$
8. $\square$ $\quad (6,1) - (7,1)$

b) Prouver que $\mathcal{S}$ est insatisfaisable :

$$\mathcal{S} = \{R, Q \vee \neg R, S \vee \neg R, P \vee \neg Q \vee \neg S, \neg P \vee \neg Q \vee \neg S\}$$

1. R	
2. $Q \vee \neg R$	
3. $S \vee \neg R$	
4. $P \vee \neg Q \vee \neg S$	
5. $\neg P \vee \neg Q \vee \neg S$	
6. Q	$(1,1) - (2,2)$
7. S	$(1,1) - (3,2)$
8. $P \vee \neg S$	$(6,1) - (4,2)$
9. P	$(7,1) - (8,2)$
10. $\neg P \vee \neg S$	$(6,1) - (5,2)$
11. $\neg P$	$(7,1) - (10,2)$
12. $\square$	$(9,1) - (11,1)$

c) $\mathcal{S}$ est-elle satisfaisable ou insatisfaisable ?

$$\mathcal{S} = \{P \vee Q, P \vee \neg Q, R \vee Q, R \vee \neg Q\}$$

1. $P \vee Q$	
2. $P \vee \neg Q$	
3. $R \vee Q$	
4. $R \vee \neg Q$	
5. P	$(1,2) - (2,2)$
6. $P \vee R$	$(1,2) - (4,2)$
7. R	$(3,2) - (4,2)$

Il n'y a pas de possibilité d'obtenir des clauses différentes, donc S satisfaisable.

d) Prouver que $\mathcal{S}$ est insatisfaisable :

$$\mathcal{S} = \{\neg P, \neg R \Rightarrow W, Q \vee (\neg T \Rightarrow \neg P \wedge \neg Q), \neg P \Rightarrow (S \wedge \neg R), \neg Q, \neg S, \neg T, \neg R \Rightarrow Y\}$$

Pour chaque formule nous en donnons sa traduction en forme clausale :

1. $\neg P$
2. $\neg R \Rightarrow W \qquad R \vee W$
3. $Q \vee (\neg T \Rightarrow \neg P \wedge \neg S) \Leftrightarrow$
$Q \vee (T \vee (\neg P \wedge \neg S)) \Leftrightarrow Q \vee ((T \vee \neg P) \wedge (T \vee \neg S)) \Leftrightarrow$
$(Q \vee T \vee \neg P) \wedge (Q \vee T \vee \neg S)$
4. $\neg P \Rightarrow (S \wedge \neg R)$
$P \vee (S \wedge \neg R) \Leftrightarrow (P \vee S) \wedge (P \vee \neg R)$
5. $\neg Q$
6. $\neg S$
7. $\neg T$
8. $\neg R \Rightarrow Y \qquad R \vee Y$

On prouve que l'ensemble de clauses est insatisfaisable :

1. $\neg P$
2. $R \vee W$
3. $Q \vee T \vee \neg P$
4. $Q \vee T \vee \neg S$
5. $P \vee S$
6. $P \vee \neg R$
7. $\neg Q$
8. $\neg S$
9. $\neg T$
10. $R \vee Y$
11. $P \qquad (5,2) - (8,1)$
12. $\square \qquad (11,1) - (1,1)$

Exercice 5

a) Donner une réfutation de l'ensemble de clauses S ci-dessous :

$$S = \{R, \neg R \vee Q, \neg R \vee S, \neg P \vee \neg Q \vee \neg S, P \vee \neg Q \vee \neg S\}$$

en utilisant la *stratégie d'entrée*.

b) La stratégie d'entrée est-elle *complète* ? Justifier.

a)

1. R
2. $\neg R \vee Q$
3. $\neg R \vee S$
4. $\neg P \vee \neg Q \vee \neg S$
5. $P \vee \neg Q \vee \neg S$
6. $\neg Q \vee \neg S$ $(4,1) - (5,1)$
7. $\neg R \vee \neg S$ $(6,1) - (2,2)$
8. $\neg S$ $(7,1) - (1,1)$
9. $\neg R$ $(8,1) - (3,2)$
10. $\square$ $(9,1) - (1,1)$

b) Non. Prendre par exemple l'ensemble de clauses *insatisfaisable* :

$$S = \{A \vee B, \neg A \vee B, A \vee \neg B, \neg A \vee \neg B\}$$

On ne pourra pas déduire $\square$ en utilisant une stratégie d'entré, parce que l'ensemble ne contient pas de clause unitaire. En effet, $\square$ ne peut être produite que par deux clauses unitaires complémentaires, dont l'une au moins doit appartenir à S (exigé par la stratégie d'entrée).

Exercice 6

Comme l'objectif dans la méthode de résolution est de détecter une contradiction élémentaire (c'est-à-dire entre deux clauses unitaires) une bonne stratégie serait d'appliquer *toujours* la règle de résolution avec au moins l'une des clauses parent unitaire (si les deux sont unitaires et la résolution s'applique, on peut arrêter, on a produit $\square$).

On appellera cette stratégie la *stratégie unitaire*.

Est-elle complète ? Justifier.

Non. Prendre le même ensemble de clauses insatisfaisable que dans l'exercice précédent :

$$S = \{A \vee B, \neg A \vee B, A \vee \neg B, \neg A \vee \neg B\}$$

On ne peut pas appliquer la règle de résolution avec une stratégie unitaire.

Exercice 7

Prouver que tout ensemble S contenant les 2^n clauses (différentes), de longueur n, que l'on peut former à partir d'un ensemble de n symboles propositionnels est insatisfaisable.

Toute interprétation I de S peut être représentée par :

$$I = \{L_1^*, L_2^*, \ldots, L_n^*\}$$

où $L_i^* = L_i$ ou $L_i^* = L_i^c$ $(1 \leq i \leq n)$ et $L_i \in Ensprop(S)$.

Puisque S contient toutes les clauses de longueur n que l'on peut former avec n symboles propositionnels, S contient une clause :

$$C_j : L_{j1} \vee L_{j2} \vee \cdots \vee L_{jn}$$

avec :

$$L_{jk} = (L_i^*)^c \ (1 \leq k \leq n)$$

Autrement sdit, si L_i est dans I, alors $\neg L_i$ est dans C_j e sir $\neg L_i$ est dans I, alors L_i est dans C_j.

C_j est falsifiée par I, donc S est falsifiée par I.

Ce raisonnement s'applique à toute interprétation de S, qui est donc insatisfaisable.

Exercice 8

a) Un ensemble de clauses ne contenant pas de clause positive ni de clause négative, peut-il être insatisfaisable ?

b) Comme corollaire de la réponse, prouver qu'un ensemble de clauses insatisfaisable contient (au moins) *une clause positive et une clause négative.*

Non. Nous en donnons deux preuves.

i) Toutes les clauses étant de la forme :

$$\bigvee_i P_i \vee \bigvee_j N_j \ (1 \leq i,\ 1 \leq j)$$

avec P_i : littéral positif et N_j : littéral négatif, les interprétations $I = \{P_i\}$ et $J = \{N_j\}$ sont des modèles de l'ensemble de clauses, qui ne peut donc être insatisfaisable.

ii) Puisque la résolution est complète pour la réfutation, on peut, sans perte de généralité, supposer que l'on essaie de détecter l'insatisfaisabilité de l'ensemble de clauses en appliquant la règle de résolution.

L'ensemble ne contient pas de clause unitaire. En effet, comme une clause uni taire contient un seul littéral et un littéral est soit positif, soit négatif, si l'ensemble contenait une clause unitaire, on violerait les hypothèses.

Toutes les clauses sont donc de la forme :

$$\neg A_1 \vee \cdots \vee \neg A_m \vee B_1 \vee \cdots \vee B_n \quad (m \geq 1,\, n \geq 1)$$

Sans perte de généralité on raisonne sur des clauses de longueur 2. L'application de la règle de résolution à $\neg L \vee P$ et $L \vee \neg Q$ (si L et P sont le même littéral, on peut éliminer la clause) donne une clause de la forme $P \vee \neg Q$. On ne pourra donc jamis obtenir des clauses unitaire complémentaires nécessaires à l'obtention de $\square$.

b) La question se pose de savoir si un ensemble de clauses contenant une clause positive (respectivement négative) mais pas de clause négative (respectivement positive) peut être insatisfaisable. La réponse est Non. Dans le premier cas, l'interprétation $l = \{P_i\}$ en est un modèle. Dans le second, l'interprétation $J = \{N_j\}$ en est un. D'où la conclusion demandée.

Exercice 9

Spécifier, en utilisant la logique propositionnelle que l'on peut colorier une carte où figurent N pays en utilisant trois couleurs différentes, de telle façon qu'en utilisant une seule couleur par pays, deux pays frontaliers n'ont jamais la même couleur (il y aura des cartes où ce coloriage n'est pas possible).

Comme $N \in \mathbb{N}$ n'est pas spécifié, on donnera un *schéma* de la spécification.

Il s'agit d'une réduction au problème SAT. Le problème de la 3-coloriabilité est donc

aussi en NP.

Quelle est la *taille* de la spécification ?

On utilise le symbole propositionnel :

P_i^j : le pays i est colorié avec la couleur j.

Chaque pays est colorié avec exactement une couleur :

$$\text{pour } 1 \leq i \leq N : (P_i^1 \wedge \neg P_i^2 \wedge \neg P_i^3) \vee (P_i^2 \wedge \neg P_i^1 \wedge \neg P_i^3) \vee (P_i^3 \wedge \neg P_i^1 \wedge \neg P_i^2)$$

En transformant sous forme clausale (en simplifiant les clauses contenant des tautologies et en appliquant la règle de subsomption) on obtient une spécification équivalente :

$$\text{(Sch. 1) pour } 1 \leq i \leq N : (P_i^1 \vee P_i^2 \vee P_i^3) \wedge (\neg P_i^1 \vee \neg P_i^2) \wedge (\neg P_i^1 \vee \neg P_i^3) \wedge (\neg P_i^2 \vee \neg P_i^3)$$

(Sch. 1) aurait aussi pu être obtenu en traduisant : « le pays i doit être colorié avec une couleur (premier conjoint) et on ne peut pas le colorier avec plus d'une (trois autres conjoints) ». La complexité de la spécification de ces clauses est inférieure à $3 \times N$, autrement dit $O(N)$.

Deux pays frontaliers n'ont pas la même couleur :

$$\text{(Sch. 2) pour tous les pays frontaliers } i, j : \neg((P_i^1 \wedge P_j^1) \vee (P_i^2 \wedge P_j^2) \vee (P_i^3 \wedge P_j^3))$$

Chaque pays a, au plus $N - 1$ voisins, donc il y aura moins de N^2 clauses comme celles ci-dessus. Donc la complexité de la spécification de cette deuxième partie est $O(N^2)$.

Au total, la complexité est donc $O(N^2)$.

Exercice 10

Un ensemble de clauses *insatisfaisable* S est dit *minimalement insatisfaisable* si et seulement si pour tout $R \subset S$ (c'est-à-dire $R \subseteq S$ et $R \neq S$), R est satisfaisable.

a) Montrer que tout sous-ensemble d'un ensemble de clauses (et plus généralement

d'un ensemble de fbf) satisfaisable est satisfaisable.

b) Montrer que tout sur-ensemble d'un ensemble de clauses (et plus généralement d'un ensemble de fbf) insatisfaisable est insatisfaisable.

c) Montrer qu'un ensemble de clauses minimalement insatisfaisable ne contient pas de littéral pur.

d) Un ensemble de clauses insatisfaisable ne contenant pas de littéral pur est-il nécessairement minimalement insatisfaisable ?

a) Soit S sat et $S_1 \subsetneq S$. S_1 ne peut pas être insat, parce que tout modèle de S est modèle de toutes les clauses de S. Donc S_1 sat.

b) Conséquence immédiate de a).

c) On suppose, sans perte de généralité, que S contient un seul littéral pur : $L \in C \in S$ et L : pur.

Soit $S_1 = S \setminus \{C\}$. Alors S_1 est sat puisque S est minimalement insatisfaisable. Soit alors M_1 un modèle de S_1. Alors $L^c \notin M_1$ puisque L est pur. Par conséquent, $M_1 \cup \{L\}$ est un modèle de S. S est donc sat. Contradiction.

d) Non nécessairement, l'ensemble de clauses :

$$\{A, \neg A, B, \neg B\}$$

ne contient pas de littéral pur, mais contient des sous-ensembles propres insatisfaisables.

Exercice 11

a) Montrer que si un ensemble de clauses S est insatisfaisable, alors *il n'existe pas* d'interprétation falsifiant *toutes* les clauses de S.

b) Soit S un ensemble de clauses insatisfaisable et $\mathcal{I}$ une interprétation pour S. Montrer qu'il existe $S_1 \subset S$, $S_2 \subset S$, $S_1 \neq \emptyset$, $S_2 \neq \emptyset$, $S_1 \cap S_2 = \emptyset$ tels que $\mathcal{I}$ est un modèle de S_1 et $\mathcal{I}$ est un contre-modèle de S_2.

Si S contient des clauses contenant des littéraux purs on peut les éliminer.

Supposons qu'il existe une interprétation $\mathcal{I}$ falsifiant *toutes* les clauses de S. Par définition de clause, $\mathcal{I}$ évalue *tous* les littéraux de C à F.

Comme nous avons éliminé de S toutes les clauses contenant des littéraux purs, si $L \in C \in S$, alors il existe $L^c \in D \in S$. L^c est évalué à V donc, par définition de clause, D à V. Contradiction. Donc $\mathcal{I}$ ne peut pas exister.

Corollaire immédiat du a).

ANNEXE 1 : LA METHODE DE RESOLUTION DANS LA LP0

La méthode de résolution est l'une des plus utilisée en pratique (surtout dans sa version pour la logique du premier ordre).

Elle utilise une seule règle d'inférence ce qui fait qu'elle est particulièrement facile à implémenter, mais elle a besoin d'une forme normale : la forme clausale (ou fnc).

Nous commençons par quelques remarques :

– La méthode exige comme format d'entrée des ensembles de clauses (fnc). Ceci n'est pas une limitation parce que toute fbf de la LP0 peut être transformée en une fbf équivalente sous fnc.

– Une clause (ou un ensemble contenant une clause) est, par définition, satisfaisable.

– $P \wedge \neg P$ n'est pas une clause, bien que cette contradiction soit représentée par ce que l'on appelle la clause vide (notée $\square$). P est une clause unitaire et $\neg P$ en est une autre.

– On considère une fbf sous forme clausale indifféremment comme une fbf sous fnc ou comme un *ensemble* de clauses et les clauses comme des *ensembles* de littéraux.

Définition 1. *Soit $S = \{C_1, \ldots, C_n\}$ un ensemble de clauses. Un ensemble de littéraux M est un modèle de S si et seulement si :*
Si $L \in M$, alors $L^c \notin M$ et :

$$C_k \cap M \neq \emptyset \quad (0 \leq k \leq n)$$

Cette définition peut être exprimée en français en disant : pour évaluer à V un ensemble de clauses il faut évaluer à V toutes ses clauses. On ne peut pas évaluer un littéral à V et à F simultanément. Pour évaluer à V une clause il faut et il suffit d'évaluer à V au moins l'un de ses littéraux.
Comme conséquence, si tous les littéraux d'une clause sont évalués à F , la clause sera évaluée à F.

Définition 2. *Soient données deux clauses contenant des littéraux complémentaires : $C_1 : L \vee \alpha$ et $C_2 : L^c \vee \beta$ (α et β sont des clauses c'est-à-dire des disjonctions de littéraux.*
La règle d'inférence nommée **règle de résolution** *est définie comme suit :*

$$R : \frac{L \vee \alpha \quad L^c \vee \beta}{\alpha \vee \beta}$$

On écrira aussi :

$$R(C_1, C_2) = C$$

ou, si l'on veut mettre en évidence les littéraux complémentaires :

$$R(C_1, C_2, L, L^c) = C$$

où

$$C = (C_1 \setminus \{L\}) \cup (C_2 \setminus \{L^c\})$$

La clause $C : \alpha \vee \beta$ est appelée **la résolvante** de C_1 et C_2.
C_1 et C_2 sont appelées **clauses parentes**.

C est une *conséquence logique* de $\{C_1, C_2\}$ (de $C_1 \wedge C_2$) mais *n'est pas* équivalente à $C_1 \wedge C_2$: tout modèle de α (resp. β) est un modèle de la résolvante, mais pas nécessairement des deux clauses parentes.

Dans le cas où α et β ne contiennent pas de littéraux :

$$\frac{L \quad L^c}{\square}$$

où $\square$, dénotant la *contradiction*, est appelé *la clause vide.*

On utilisera également la règle dite *d'absorption* :

$$Abs : \frac{\alpha \vee L \vee \beta \vee L \vee \beta \vee \gamma}{\alpha \vee L \vee \beta \vee \gamma}$$

où α, β, γ sont des clauses.

Cette règle revient à considérer les clauses comme des ensembles de littéraux ou, ce qui est équivalent, à utiliser les propriétés d'associativité, commutativité et d'absorption du $\vee$: $(\alpha \vee \beta) \vee \gamma \Leftrightarrow \alpha \vee (\beta \vee \gamma)$, $L \vee \alpha \Leftrightarrow \alpha \vee L$, $L \vee L \Leftrightarrow L$ respectivement.

Remarque importante. On ne doit pas confondre la clause vide avec un ensemble de clauses vide. La première est insatisfaisable et le second est satisfaisable (il ne contient rien, il ne peut donc pas contenir une contradiction).
On peut en donner une preuve plus formelle. Par l'absurde : si $\emptyset$ est insatisfaisable, alors (par exemple) $\{A \vee B\} = \emptyset \cup \{A \vee B\}$ serait insatisfaisable, puisque tout sur-ensemble d'un ensemble contradictoire est lui-même contradictoire. Mais $\{A \vee B\}$ est satisfaisable (modèles $\{A\}$, $\{B\}$, $\{A, B\}$) : contradiction. Donc $\emptyset$ est satisfaisable.

Pour une règle non-déterministe comme la résolution il est utile de définir un *opérateur* permettant de capturer toutes les résolvantes que l'on peut obtenir en **appliquant la règle de résolution de toutes les façons possibles**.

Définition 3 (opérateur $\mathcal{R}$). *Soit S un ensemble fini de clauses. On définit :*

$$\begin{aligned}
\mathcal{R}(S) &= S \cup \{R(C_1, C_2) | C_1, C_2 \in S\} \\
\mathcal{R}^0(S) &= S \\
\mathcal{R}^{n+1}(S) &= \mathcal{R}(\mathcal{R}^n(S)) \quad (n \geq 0) \\
\mathcal{R}^*(S) &= \bigcup_{n \geq 0} \mathcal{R}^n(S)
\end{aligned}$$

Remarque. Pour un ensemble fini satisfaisable de clauses propositionnelles S, il existe n tel que

$$\mathcal{R}^*(S) = \mathcal{R}^n(S)$$

(voir l'exercice 3 de ce TD).

Définition 4 (un système déductif pour la résolution).

$\mathcal{S}_R =< \mathcal{L}, \mathcal{R}, \mathcal{A} >$

$\mathcal{L}$: clauses et ensembles de clauses

$\mathcal{R} = \{R, Abs\}$

$\mathcal{A} = \emptyset$

On dit que la clause C **se déduit par résolution** de l'ensemble de clauses S, noté :

$$S \vdash_{\mathcal{S}_R} C$$

ou

$$S \vdash_{\mathcal{R}} C$$

si et seulement s'il existe une suite finie $C_1, \ldots, C_k$ telle que :

$$\begin{aligned}
C_k &= C \\
C_{i+1} &= R(C_m, C_n) \quad (0 \leq i \leq k-1) \\
C_m, C_n &\in S \cup \{C_1, \ldots, C_i\}
\end{aligned}$$

La suite $C_1, \ldots, C_k$ s'appelle une **déduction** à partir de S.

Si $C = \square$, elle s'appelle une **réfutation** de S.

Pour la méthode de résolution, la correction et la complétude (pour la réfutation) s'énoncent :

– Correction : si $S \vdash_{\mathcal{R}} \square$ alors S est insatisfaisable (contradictoire) ;

– Complétude pour la réfutation : Si S est insatisfaisable (contradictoire), alors $S \vdash_{\mathcal{R}} \square$.

L'expression *complétude pour la réfutation* appliquée à la méthode de résolution s'explique facilement en constatant, par exemple, que :

$$A \nvdash_{\mathcal{R}} A \vee B$$

bien que $A \vee B$ soit une conséquence logique de A.

En revanche, si l'on nie $A \vee B$, on obtient l'ensemble de clauses $\{A, \neg A, \neg B\}$ et immédiatement la clause $\square$ en appliquant la résolution entre A et $\neg A$.

Théorème 1. *Soit S un ensemble satisfaisable de clauses et M un modèle de S. Si $S \vdash_{\mathcal{R}} C$, alors $M \cap C \neq \emptyset$.*

Preuve. Si C est obtenu par application de la règle de résolution, alors il existe $C_1 \in S$, $C_2 \in S$ et un littéral $L \in C_1$ tel que $L^c \in C_2$ et

$$\begin{aligned} C &= R(C_1, C_2, L, L^c) \\ &= (C_1 \setminus \{L\}) \cup (C_2 \setminus \{L^c\}) \end{aligned}$$

Puisque M est un modèle de S, M est un modèle de toutes les clauses de S, donc :

$$M \cap C_1 \neq \emptyset \text{ et } M \cap C_2 \neq \emptyset$$

Trois cas sont à considérer :

i) Si $L \in M$ et puisque M est un modèle de C_1 et de C_2, il existe $K \in C_2 \setminus \{L^c\}$ tel que $K \in M$. Donc, par définition de la règle R : $K \in C$, donc $M \cap C \neq \emptyset$.

ii) Si $L^c \in M$, alors il existe $N \in C_1 \setminus \{L\}$ et $N \in M$.

Par définition de la règle R, $N \in C$, donc $M \cap C \neq \emptyset$.

iii) Si $L \notin M$ et $L^c \notin M$, alors M ne dépend pas des valeurs données à L et à L^c, donc $M \cap C \neq \emptyset$.
La preuve est complétée en appliquant la définition de déduction et par induction sur la longueur de la déduction. ■

On peut énoncer ce théorème avec une autre notation : *Si* $C \in \mathcal{R}^*(S)$ *et* $\models_{\mathcal{I}} S$ *alors* $\models_{\mathcal{I}} C$.
Nous avons utilisé la contraposée, c'est-à-dire :

$$\text{si } \not\models_{\mathcal{I}} C \text{ alors } \not\models_{\mathcal{I}} S$$

pour fermer des branches dans les arbres sémantiques. Elle est aussi utilisée dans des solveurs SAT (c'est-à-dire des logiciels permettant de résoudre le problème SAT) très performants pour élaguer l'espace de recherche. En effet, quand on vérifie que l'interprétation (en général partielle, mais qui peut suffire à évaluer certaines clauses) proposée falsifie une clause (d'entrée ou déduite par résolution) , ce n'est pas la peine de continuer dans cette voie.

Corollaire 1 (correction de la résolution). *La méthode de résolution est correcte.*

Preuve. Si $S \vdash_{\mathcal{R}} \square$ et S satisfaisable, alors $\square$ serait satisfaisable d'après le théorème 1. Or c'est impossible. Donc, S est insatisfaisable. ■

Exemple. Il s'agit de trouver, en utilisant la méthode de résolution que l'ensemble de clauses :

$$S = \{\neg P \vee \neg Q \vee R, P \vee R, Q \vee R, \neg R\}$$

est insatisfaisable et ceci en vue de concevoir dans le futur un *programme* pour le faire.

Le problème principal qui se présente à nous est celui de gérer le *non-déterminisme* (c'est-à-dire les *choix* pour l'application de la règle de résolution).

Avant d'analyser les bons choix d'application de la règle et pour être sûrs que la méthode marchera dans tous les cas, on décide d'appliquer la « méthode de la force brute », c'est-à dire qu'on applique *tous les choix* pour une certaine énumération et l'on regarde si l'on a (dans l'ensemble des clauses du problème auquel on ajoute les clauses déduites) deux clauses unitaires complémentaires (la seule contradiction que l'on puisse toujours détecter de façon mécanique).

La notation à droite des formules :

$$(i,j)-(k,l) \quad (1 \leq i \leq 4, 1 \leq j \leq 3, 2 \leq K \leq 12, 1 \leq l \leq 2)$$

signifie : on applique la règle de résolution en choisissant le littéral à la position j (de gauche à droite) de la clause numérotée i et son complémentaire à la position l de la clause numérotée k.

1	$\neg P \vee \neg Q \vee R$	
2	$P \vee R$	
3	$Q \vee R$	
4	$\neg R$	
5	$\neg Q \vee R$	$(1,1)-(2,1)$
6	$\neg P \vee R$	$(1,2)-(3,1)$
7	$\neg P \vee \neg Q$	$(1,3)-(4,1)$
8	P	$(2,2)-(4,1)$
9	Q	$(3,2)-(4,1)$
10	$\neg Q \vee R$	$(1,1)-(8,1)$
11	$\neg P \vee R$	$(1,2)-(9,1)$
12	R	$(2,1)-(6,1)$
13	$\square$	$(4,1)-(12,1)$

On remarque que l'on peut déduire la même clause plus d'une fois (par exemple 5 et 10 ; 6 et 11).

En analysant la réfutation après l'avoir obtenue, on constate que seules 6 et 12 étaient nécessaires pour détecter la contradiction. Vous semble-t-il possible de savoir qu'il y a contradiction avant de trouver la réfutation ? ■

ANNEXE 2 : PROBLEMES, STRATEGIES ET ENONCES

Nous avons vu différentes procédures de preuve appelées aussi calculs (systèmes formels ou « à la Hilbert », tableaux, résolution, etc.). La question se pose alors naturellement : « existe-t-il une procédure de preuve qui soit uniformément (c'est-à dire pour tout problème) meilleure (par exemple en nombre de pas) que les autres ? » . La réponse (à laquelle on peut s'attendre) est non.

Une notion naturellement associée au problème du non-déterminisme est celle de *stratégie*. Ce mot a, au sens technique, une signification très similaire à celle qu'il a dans le langage courant.

Une stratégie est une règle pour faire les choix d'application d'une(des) règle(s) d'inférence non-déterministe(s) dans le but d'obtenir un certain objectif, en général, en diminuant le nombre de choix , donc l'espace de recherche (c'est-à-dire l'ensemble de toutes les applications avant d'arriver au résultat cherché ou d'arrêter). Parfois on cherche à diminuer le nombre des pas menant à la solution.

Dès le début on s'est rendu compte de l'impossibilité d'attaquer des problèmes intéressants de preuves automatisées sans associer des stratégies aux calculs (les calculs étant des ensembles de règles d'inférence, règles non-déterministes).

Bien entendu, on s'est posé la question concernant l'existence de la meilleure façon de gérer le non-déterminisme, via une procédure (stratégie) parfaite, c'est-à dire n'engendrant jamais de formule redondante et ce pour la résolution de n'importe quel problème.

On peut montrer (en utilisant des résultats connus de la théorie de la calculabilité) qu'il n'existe pas de procédure complète pour la réfutation (par exemple la résolution)

parfaite, c'est-à-dire n'engendrant pas de formule (clause) non nécessaire à la preuve (réfutation).

On peut définir de façon abstraite une *procédure de preuve* comme un couple (T, Σ) où T est un système formel et Σ une stratégie pour T.

Il est intéressant de remarquer qu'en général dans les ouvrages de logique on parle de *systèmes de preuve (proof systems)* en les identifiant à T et sans faire mention de la stratégie.

Dans le but de définir la notion abstraite de *preuve automatisée*, on définit la notion de *graphe de preuve*, qui suit naturellement la définition de l'opérateur $\mathcal{R}$: Les formules ont un niveau, qui n'est pas unique (d'où l'utilisation de graphe au lieu d'arbre) et qui est défini de façon standard comme 1 de plus que celui des formules dont elle est la conséquence directe. Autrement dit, si l'on utilise la résolution, le niveau de la résolvante est 1 de plus que celui de ses parents (les clauses d'entrée ont le niveau 0).

Le *problème du démonstrateur* pour un triplet :

$$(S_0, \Gamma, F)$$

est défini comme celui d'engendrer en utilisant une stratégie Σ un ensemble de formules F, avec :
S_0 : ensemble de formules de départ ;
Γ : ensemble des règles d'inférence et

$$\Gamma^*(S_j) = \bigcup_{i \geq j} \Gamma(S_i)$$

F : ensemble de formules sous-ensemble des conséquences de S_0, c'est-à-dire que $F \subseteq \Gamma^*(S_0)$;

et :

$$\Sigma : 2^G \to 2^G$$

où G est le graphe de preuve.

En dépliant le graphe sous forme d'arbre on associe à chaque nœud une dérivation, ce qui permet d'associer à une feuille une mesure de la dérivation avec la stratégie Σ.

Nous pouvons modifier légèrement la définition pour permettre de caractériser aussi les assistants de preuve, en particulier les vérificateurs de preuve : Un *démo-vérificateur abstrait* est un 5-uplet :

$$(S_0, \Gamma, F, P, \Sigma)$$

où P est l'ensemble des formules de la prétendue preuve (si $P = \emptyset$, on a un démonstrateur totalement automatisé, si P contient tous les pas d'une preuve nous avons un vérificateur, si l'on en donne, par exemple, certains lemmes on a un assistant de preuve ou démonstrateur interactif).

Nous avons inclus la stratégie qui n'est pas nécessairement uniforme : on peut penser Σ comme un ensemble de stratégies. La théorie dans laquelle la preuve est faite est incluse dans S_0.

Définition 1. *Une stratégie st pour la résolution est dite* **complète** *si et seulement si pour tout ensemble de clauses S :*

$$\textit{Si } S \vdash_{\mathcal{R}} \square \textit{ alors } S \vdash_{\mathcal{R}+st} \square$$

Exemple (stratégie d'entrée). Un exemple de stratégie pour la résolution est la *stratégie d'entrée* : étant donné un ensemble de clauses S, la résolution est appliquée toujours avec au moins une clause dans S (l'ensemble de clauses d'entrée).

Définition 2 (complexité d'une preuve, d'une méthode). *La complexité d'une preuve (réfutation) par résolution d'un ensemble de clauses S, notée $Comp_{\mathcal{R}}(S)$ est le nombre de clauses différentes de la preuve (réfutation) de S.*
La complexité de la méthode de résolution sur des ensembles de clauses de cardinalité n, notée $Comp_{\mathcal{R}}(n)$ est définie comme :

$$Comp_{\mathcal{R}}(n) = \max_{card(S)=n} \min Comp_{\mathcal{R}}(S).$$

Le problème de la complexité des preuves de la LP0 a été particulièrement étudié depuis la fin des années 1960.